UNIVERSITY OF ALBERTA

Optoelectronic Fast Packet Switching

by

Robert Tholl ©

A thesis submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science.

Department of Electrical Engineering

Spring 1997

National Library of Canada
Acquisitions and Bibliographic Services
395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale du Canada
Acquisitions et services bibliographiques
395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence

Our file Notre référence

The author has granted a non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced with the author's permission.

L'auteur a accordé une licence non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

0-612-21213-0

Canada

UNIVERSITY OF ALBERTA

RELEASE FORM

Name Of Author: Robert Tholl

Title Of Thesis: Optoelectronic Fast Packet Switching

Degree: Master of Science

Year This Degree Granted: Spring 1997

Permission is hereby granted to the University of Alberta to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves all other publication and other rights in association with the copyright in the thesis, and except as hereinbefore provided neither the thesis nor any substantial portion thereof may be printed or otherwise reproduced in any material form whatever without the author's prior written permission.

Robert Tholl
1411 20 St. NW
Calgary, Alberta
T2N 2K7

April 1, 1997

UNIVERSITY OF ALBERTA

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled Optoelectronic Fast Packet Switching submitted by Robert Tholl in partial fulfillment of the requirements for the degree of Master of Science.

Dr. R. I. MacDonald

Dr. W. Grover

Dr. W. Allegretto

March 19, 1997

Dedication

To Carol, who tolerated me while I was writing.

To my parents for setting the example.

To Shanahan for learning almost quickly enough not to touch.

Abstract

Fast packet switching is introduced and briefly described, using Asynchronous Transfer Mode (ATM) as a concrete example. Fast packet switching architectures are presented and classified based on the buffering used, the signal distribution scheme used, and the switching technology used within the switch core. From this, a receiver-switched optoelectronic switching technology was chosen that uses fiber delay lines as buffers. Some physical switch architectures are simulated to determine their packet loss probability and latency characteristics. These performance comparisons are used to select a reflex switch architecture with single packet buffers and an internal switch core speedup. Finally, the components used to build a fast packet switch are described and characterized, either through experimental results or from results in the literature.

Acknowledgments

I would like to thank my supervisor, Dr. Ian MacDonald, for allowing me to follow my interests in this project, even when those interests were not in tune with his research directions. I must also thank Rohit Sharma, Barrie Keyworth, Jim Slevinsky, Dino Corazza, and Rainer Iraschko for their support and assistance in the various stages of this project. David Clegg deserves special mention for sharing both his deep understanding of radio frequency design and construction, as well as his hands-on laboratory experience.

I would also like to thank Julian Noad of CRC and Graham McKinnon of AMC for their generous assistance in fabricating and assembling some of the components used in this project.

I am thankful for the financial assistance provided by both NSERC and TRLabs.

I am grateful to the staff and students of both the Electrical Engineering department and TRLabs for their help, support, and comradeship. This is especially true for the members of the softball team and the golf tournament players. I would also like to thank Dr. Ron Lawson, the graduate student co-ordinator for the electrical engineering department, for his unfailing support.

Finally, I wish to extend my thanks to the examining committee members for reviewing my work and making constructive suggestions.

Table of Contents

Introduction 1
Ideal Switch Architectures 11
- Switch Buffering Classification 12
  - Input Buffering 13
  - Simple Output Buffering 14
  - Shared Output Buffering 15
  - Route Buffering 16
- Switch Core Classification 18
  - Time Division Multiplexing - Common Bus 19
  - Time Division Multiplexing - Shared Memory 21
  - Space Division Multiplexing - Signal Routing 23
  - Space Division Multiplexing - Signal Distribution 27
  - Wavelength Division Multiplexing - Multihop Switches 29
  - Wavelength Division Multiplexing - Wavelength Agile Components 31
- Switch Core Technology 34

Physical Switch Architectures 39
- Crossbar Switch 41
- Knockout Switch 46
- Staggering Switch 51
- Reflex Switch - Multiple Packet Buffers 59
- Reflex Switch - Single Packet Buffers 68
- Reflex Switch - Fast Switch Core with Single Packet Buffers 73
- Switch Architecture Performance Comparisons 78

Switch Components 86
- Packet Framing and Clock Recovery 87
- Address to Output Decode 90
- Packet Synchronization 90
- Electronic Amplifiers 92
- Laser Sources 101
- Optical Signal Distribution 109
- Fiber Delay Lines 110
- Photodetector Arrays 112
- Address Replacement 122
- Simple Output Queues 122
- Controller Design 123

Conclusions 126
- Switch Architectures 126
- Switch Simulations 127
- Switch Components 127
- Future Work 128

Bibliography 129
Appendix A - Sample Simulation Input and Output Files 137
Appendix B - Crossbar Switch Theoretical and Simulated Performance 148
Appendix C - Knockout Switch Theoretical Performance 152
Appendix D - Amplifier Noise Level Calculations 157

List of Tables

Table 3.1 - Crossbar Switch Maximum ρ for Desired PLP....46
Table 3.2 - Knockout Switch PLP Change with Output Queue Size....51
Table 3.3 - Fast Core Single Packet Buffer Reflex Switch PLP Change with Output Queue Size....78
Table 4.1 - RF Amplifier Measured Performance....93
Table 4.2 - Power Amplifier Measured Performance....98
Table 4.3 - 1 to 10 Optical Power Splitter Performance....109
Table B.1 - Crossbar Switch PLP - Theory and Simulated per Batch....150
Table B.2 - Crossbar Switch PLP - Theory and Simulated per Output....151

List of Figures

Figure 1.1 - ATM Cell Format....5
Figure 1.2 - ATM Switch Block Diagram....7
Figure 2.1 - Queue Locations in a Logical Switch Architecture....13
Figure 2.2 - Optimal PLP and Latency for Various Queuing Methods [Mel92]....17
Figure 2.3 - Common Bus TDM Switch....19
Figure 2.4 - Common Memory TDM Switch....22
Figure 2.5 - Banyan and Batcher-Banyan Routing SDM Switches....24
Figure 2.6 - Signal Distribution SDM Switch....28
Figure 2.7 - Multihop WDM Switch....29
Figure 2.8 - Wavelength Agile WDM Switch....32
Figure 2.9 - Switch Core Technologies....35
Figure 2.10 - Optoelectronic Receiver-Switched Switch Core....36
Figure 3.1 - Switch Components....41
Figure 3.2 - Crossbar Switch Block Diagram....42
Figure 3.3 - Crossbar Switch Control Algorithm....43
Figure 3.4 - Theoretical and Simulated PLP Performance for Crossbar Switch....44
Figure 3.5 - The Knockout Switch Block Diagram....47
Figure 3.6 - Knockout Switch Control Algorithm....48
Figure 3.7 - Knockout Switch Performance with 10 Packet Output Queues....49
Figure 3.8 - Knockout Switch Performance with 21 Packet Output Queues....50
Figure 3.9 - Optoelectronic Staggering Switch block diagram....52
Figure 3.10 - Simple Staggering Switch Control Algorithm....54
Figure 3.11 - Out of Sequence Packets with Staggering Switch....55
Figure 3.12 - Sequence Preserving Staggering Switch Control Algorithm....57
Figure 3.13 - Simple and Sequence Preserving Staggering Switch Performance....58
Figure 3.14 - Staggering Switch Performance with N = M, various P....59
Figure 3.15 - Optoelectronic Reflex Switch Block Diagram....60
Figure 3.16 - Non-Sequence Preserving Staggered Buffer Reflex Switch Control Algorithm....62
Figure 3.17 - Sequence Preserving Staggered Buffer Reflex Switch Control Algorithm....63
Figure 3.18 - Out of Sequence Packets with Staggered Buffer Reflex Switch....64
Figure 3.19 - Simple and Sequence Preserving Staggered Buffer Reflex Switch Performance....66
Figure 3.20 - Staggered Buffer Reflex Switch Performance, Varying P....67
Figure 3.21 - Simple Reflex Switch Block Diagram....69
Figure 3.22 - Simple Sequence Preserving Reflex Switch Control Algorithm....70
Figure 3.23 - Simple Reflex Switch Performance, Varying P....72
Figure 3.24 - Simple Fast Core Optoelectronic Reflex Switch....74

Figure 3.25 - Simple Fast Core Reflex Switch Performance, Q_O = 10 Packets 75
Figure 3.26 - Simple Fast Core Switch Performance, Q_O = 21 Packets 77
Figure 3.27 - Switch Performance Comparison with Similar Number of Buffer Spaces 80
Figure 3.28 - Switch Performance Comparison with the Same Number of Crosspoints 82
Figure 3.29 - Switch Performance Comparison with the Same Number of Buffers 84
Figure 4.1 - Switch Components 87
Figure 4.2 - RF Amplifier Schematic 93
Figure 4.3 - Bandwidth Measurement Test Setup 94
Figure 4.4 - RF Amplifier Measured Bandwidth 94
Figure 4.5 - Noise Measurement Test Setup 95
Figure 4.6 - Spectrum Analyzer Measured Noise Floor 96
Figure 4.7 - RF Amplifier Measured Noise 97
Figure 4.8 - Power Amplifier Schematic 98
Figure 4.9 - Power Amplifier Measured Bandwidth 99
Figure 4.10 - RF & Power Amplifier Measured Bandwidth 100
Figure 4.11 - RF & Power Amplifier Measured Noise 101
Figure 4.12 - Laser RF Drive Circuit 103
Figure 4.13 - Laser Measured Bandwidth Test Setup 104
Figure 4.14 - Laser Measured Bandwidth - No Equalization 104
Figure 4.15 - Laser Bandwidth - Varying Poptical 105
Figure 4.16 - Laser S_{11} Impedance 106
Figure 4.17 - Laser RF Circuit - with C_{e1}, L_e, C_{e2} Equalization 107
Figure 4.18 - Laser Bandwidth - C_{e1}, L_e, C_{e2} Equalization 107
Figure 4.19 - Laser RF Circuit - with C_{e1}, R_e, L_e, C_{e2} Equalization 108
Figure 4.20 - Laser Bandwidth - C_{e1}, R_e, L_e, C_{e2} Equalization 108
Figure 4.21 - MSM Photodetector - a) Structure, b) Schematic 113
Figure 4.22 - MSM Photodetector Array - Direct Bias 114
Figure 4.23 - MSM Photodetector Array - Switched Bias 114
Figure 4.24 - MSM Photodetector Test Setup 115
Figure 4.25 - MSM Photodetector Array Bandwidth 115
Figure 4.26 - ITO MSM Photodetector Array Bandwidth 116
Figure 4.27 - MSM #1 of 8 Photodetector Array Bandwidth 117
Figure 4.28 - MSM #7 of 8 Photodetector Array Bandwidth 118
Figure 4.29 - MSM Photodetector Array Isolation ([Lam94]) 119
Figure 4.30 - MSM Photodetector Array Isolation (similar to [Gou94]) 120
Figure 4.31 - ATM Cell Switching Time 120
Figure B.1 - Crossbar Switch Block Diagram 148
Figure B.2 - Crossbar Switch PLP - Theory minus Simulated 151
Figure C.1 - Knockout Switch Block Diagram 152
Figure C.2 - Knockout Switch State Transition Diagram 154
Figure D.1 - Spectrum Analyzer Noise Floor Test Setup 157
Figure D.2 - RF Amplifier Noise Level Test Setup 158
Figure D.3 - Noise Level RF Amp & Spectrum Analyzer Test Setup 159
Figure D.4 - Power Amplifier Noise Level Test Setup 160

List of Nomenclature

AAL	ATM adaptation layer. There are several defined methods for using the 48 byte ATM cell payload to carry traffic based on other standards (such as PCM encoded voice) to meet different service requirements.
µsec	microsecond = 10^{-6} second.
ρ	offered load to an input (the probability a packet arrives at an input in a single input time slot), assumed to be uniform.
ASIC	Application specific integrated circuit.
ATM	Asynchronous transfer mode, an international telecommunications standard for cell switched networks.
BER	Bit error rate.
blocking	Given a switch with N inputs and M outputs, it may not be possible for min(N,M) input and output pairs to be connected simultaneously, even if all the pairs are disjoint. In simpler terms, given a free input and a free output, it is not always possible to make a connection between the two.
broadcast	A packet distribution scheme where a single packet is sent to multiple destinations. The packet is only replicated when the paths to the destinations diverge.
CCITT	See ITU.
cell	An ATM packet, that is a packet that is 53 bytes long and follows the format for ATM cells as defined by the ITU.
cell loss probability (CLP)	Probability of a cell that arrived at an input to the ATM switch not exiting the switch at an output, i.e. the cell is lost within the switch.
core time slot	A time slot based on the bit rate of the switch core.
CRC	Cyclic redundancy check.
crosspoint	A switching element in the switch core used to connect a switch inlet to a switch outlet in some fashion.
dB	Decibel.
DC	Direct current.
E/O	Electrical to optical conversion, i.e. a laser.
FET	Field effect transistor.
FIFO	First in - first out, a memory that stores information in the order it was received and forces the information to be read out in the same order.
GaAs	Gallium arsenide.

Gbps	Gigabits per second = 10^9 bits per second.
GBps	GigaBytes per second = 10^9 bytes per second = 8 Gbps.
Gcps	Gigacells per second = 10^9 ATM cells per second ~ 53 GBps.
GHz	Gigahertz = 10^9 Hz.
Gpps	Gigapackets per second = 10^9 packets per second.
incast	A connection that sends the packets from more than one switch input to a single switch output. Also known as a multipoint to point connection.
inlet	Any port on a switch that has packets arriving to the core of the switch.
input	A port on a switch that has packets arriving to the core of the switch from the network, not from buffers used within the switch.
input time slot	A time slot based on the bit rate of the inputs.
ITO	Indium Tin Oxide, a metal transparent to 800 nm wavelength light that is used in fabricating MSM photodetectors.
ITU	International Telecommunications Union, an international standards setting body (formerly called CCITT).
L	The number of packets that can be delivered to a single output in a single input time slot for a given switch architecture.
LAN	Local area network.
latency	The amount of delay, usually measured in time slots, that a packet experiences in passing through a switch.
km	kilometer.
M	The number of outputs from a switch.
Mbps	Megabits per second = 10^6 bits per second.
MBps	MegaBytes per second = 10^6 bytes per second = 8 Mbps.
Mcps	Megacells per second = 10^6 ATM cells per second = 53 MBps.
Mpps	Megapackets per second = 10^6 packets per second.
MSM PD	Metal-semiconductor-metal photodetector.
multicast	A connection that sends a single packets from one switch input to more than one switch output. Also known as a point to multipoint connection.

N	The number of inputs to a switch.
non-blocking	Given a switch with N inputs and M outputs, it is possible for min(N,M) input and output pairs to be connected simultaneously, provided that all the pairs are disjoint. In simpler terms, given a free input and a free output, it is always possible to make a connection between the two.
NNI	Network node interface, the format used for an ATM cell header when being routed on a link within the ATM network.
nsec	nanosecond = 10^{-9} seconds.
O/E	Optical to electrical conversion, i.e. a photodetector.
OC-n	SONET defined optical communication rate where n is the multiple of the basic OC-1 rate of 51.840 Mbps. Common rates are OC-3 (155.520 Mbps), OC-12 (622.080 Mbps), OC-24 (1.244160 Gbps), and OC-192 (9.95328 Gbps).
outlet	Any port on a switch that has packets exiting from the core of the switch.
output	A port on a switch that has packets exiting from the core of the switch to the network, not into buffers used in the switch.
output time slot	A time slot based on the bit rate of the outputs.
P	The number of buffers (usually shared) in a switch.
packet	A group of information being sent from one location to another, used in this thesis for fixed length groups of information (all the same size).
packet loss probability (PLP)	Probability of a packet that arrived at an input to the switch not exiting the switch at an output, i.e. the packet is lost within the switch.
PCM	Pulse coded modulation, a method for sampling an analog signal and converting it to a binary encoded value for transmission over a digital network.
psec	picosecond = 10^{-12} seconds
Q_i	Size of the input queue (in packets).
Q_o	Size of the simple output queue (in packets).
Q_s	Size of the total shared output queue (in packets).
RAM	Random access memory.
rearrangeably non-blocking	Given a switch with N inputs and M outputs, it is possible for min(N,M) input and output pairs to be connected simultaneously, provided that all the pairs are disjoint. Connecting the input and output pair may involve rearranging the internal switch connections of other pairs.
RF	Radio frequency.

ROM	Read only memory.
SDM	Space division multiplexing.
signal bandwidth	The bandwidth of a signal that can be transmitted through a switching element without being significantly altered (i.e. ignoring added noise).
single-cast	A connection that sends a single packets from one switch input to a single switch output. Also known as a point to point connection.
SNR	Signal to noise ratio.
SONET	Synchronous Optical NETwork, an ITU defined international standard for point to point optical communications at various bit rates (see OC-n).
square switch	A switch that has the same number of inputs and outputs.
strictly non-blocking	Given a switch with N inputs and M outputs, it is possible for min(N,M) input and output pairs to be connected simultaneously, provided that all the pairs are disjoint. Connecting the input and output pair does not involve rearranging the internal switch connections of other pairs.
switch core	The portion of a switch that connects the inlets to the outlets with all buffering actions excluded.
switching bandwidth	The rate at which a switching element can change from one input connected to an output to a different input connected to the output.
TCP/IP	Transmission control protocol/Internet protocol, the two standard protocols used for transferring information across the Internet.
TDM	Time division multiplexing.
time slot	In fixed length packet switches, the smallest division of time is the length of time taken to transmit one packet at the given bit rate is termed a time slot.
UNI	User network interface, the format used for an ATM cell header when being routed on a link entering or exiting the ATM network.
unicast	A connection that sends packets from a single switch input to a single switch output. Also known as a point to point connection.
VCI	Virtual Circuit Identifier, bits 13 - 28 of the ATM cell header used to indicate one circuit out of a group routed along the same path.
VCI/VPI	The routing information contained in the ATM cell header, bits 1 - 28.
VLSI	Very Large Scale Integrated Circuit.
VPI	Virtual Path Identifier, bits 1 - 12 of the NNI ATM cell header or bits 5 - 12 of the UNI ATM cell header used to indicate the path this cell should be routed along.
WDM	Wavelength division multiplexing.

X.25 An early international standard protocol for packet switched data communication.

Introduction

The Asynchronous Transfer Mode (ATM) international standard is an example of the fast packet class of electronic communication standards. The fast packet communication standards use fixed length packets to transfer information from one site to another and perform error and flow control at the end nodes as opposed to within the network. The fixed length packets allow for much simpler switching and routing, enabling higher bit rates within the network. Moving the error and flow control functions from within the network, at every node, to the periphery of the network, at the originating and destination nodes, also simplifies each internal node within the network, again allowing for higher bit rates within the network. The other advantage of fast packet communications, the standard advantage of packet based communication systems, is that multiple communication channels can efficiently share a single communication link with each channel receiving the bandwidth it requires at that specific time. This ability of fast packet communications to share the available bandwidth efficiently among many different communication channels has resulted in a tremendous amount of interest, particularly in the area of ATM communications. As discussed next, the ATM communications standard allows the integration of many communication traffic types onto one network, allowing for economies of scale in the operation of the resulting network.

The telegraph, one of the original telecommunication systems, was a message based service that shared the communications medium among many users. The telegraph sent a single complete message at a time, with each message using the entire bandwidth of the communication medium available, from one relay point to the next. The human operator then routed the message to the next relay point towards its destination. The next message was then

transmitted using the entire bandwidth of the communications medium for the duration of the message. It is interesting to note that the telegraph network sent symbols consisting of long and short pulses, a form of digital transmission.

The telephone fundamentally changed the nature of the telecommunication network. The most visible change to the user was the conversion to a full duplex real time voice connection between the communicating parties. This allowed untrained users to communicate, rather than requiring a trained operator to encode and decode the message at each end. Communicating by voice, rather than long and short pulses, requires an analog connection with a high enough signal to noise ratio (SNR) for the voice to be decipherable. Modern telephone systems have quantized the analog voice signal and pulse code modulated (PCM) the resulting signal to allow digital transmission of the analog voice signal. To allow a full duplex real time connection between the calling partners the telephone network uses circuit switching, where a connection between the parties is established when the phone call is made and the connection lasts until the phone call is finished. The connection reserves resources within the telecommunications network for the duration of the call. If the required resources are not available the call is not completed, resulting in a busy signal. Originally, a human operator was responsible for establishing and freeing the connections, although electro-mechanical switches (such as the Strowger switch) were soon introduced. These electro-mechanical switches were then superseded by electrical and totally electronic switches. The resulting telephone network has an architecture designed to support very many communication channels of relatively low bandwidth that have equal traffic loads in both directions. Each call has an average duration of a few minutes, and there are very many possible originating and receiving sites. Long delays in the data transmission, on the order of ten

milliseconds or more, result in intolerable echoes in the conversation and must be avoided. In fact, moderate data loss, resulting in distortion of the voice signal, is preferable to extra delay [Fre89]. Long delays, on the order of tens of seconds, are tolerable in setting up the call.

Cable television networks and computer communications have become increasingly important recently. The traffic patterns of these networks are very different from each other, as well as different from the telephone network traffic patterns. For example, a cable television network distributes a number of high bandwidth analog signals from a central site to many receiving sites for long durations. The television signal is half duplex, that is, the signal travels in one direction only. This results in a cable television network using a star architecture that distributes the same television signals from one central site to all the receiving sites. Here, the amount of delay variation is critical, and the data must be delivered with delay variations of milliseconds to achieve acceptable performance.

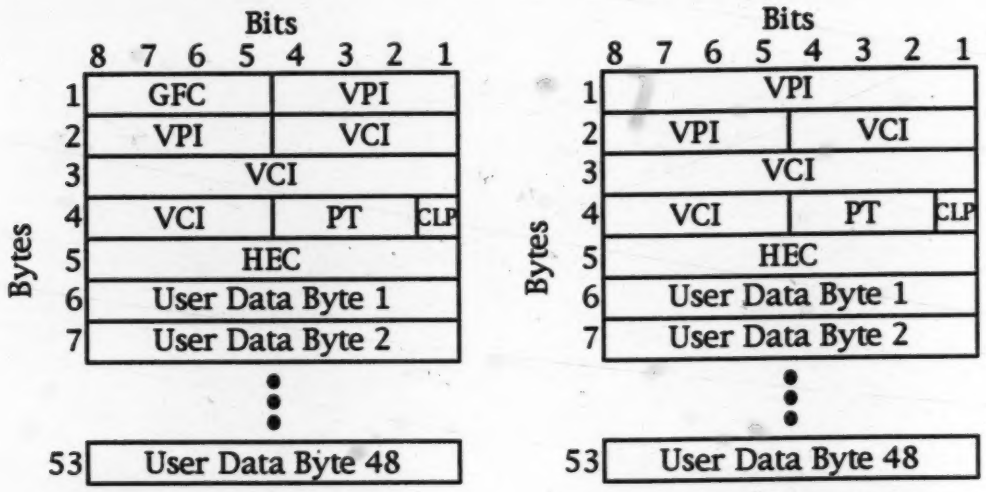
On the other hand, data communications between computers involves a bursty form of traffic, with varying bandwidths required depending on the demands of the moment. The traffic flow varies from very little or none for long periods to large amounts of data that should be transferred as quickly as possible with absolutely no errors. The lower the delay, the better the network performance, but data integrity is much more important than the amount of delay experienced for most forms of data communications. Data communications used in real-time processing or distributed processing may have strict upper limits on the tolerable delay, usually in the 50 millisecond range [dPr91].

In recent years, the total volume of communications traffic has been increasing rapidly. Of that traffic load, by far the largest growth is occurring in data communications, as evidenced by the current interest in the Internet.

Predictions range from a tenfold increase in traffic by the year 2000 [Cra95] to the current doubling of the data traffic on the Internet every twelve months [Com96]. Data communications, such as computer to computer traffic, forms the majority of the other traffic currently, although video communication traffic is expected to increase rapidly. For example, one hour of an uncompressed high definition television (HDTV) show requires approximately 540 GBytes of storage, which reduces to about 9 GBytes when the show is compressed [Jai94]. Assuming constant bit rate traffic for the compressed show gives a traffic load for a single compressed HDTV show of 2.5 MBytes per second (MBps). In comparison, a single voice call generates a traffic load of 8 kBps, so a single compressed HDTV show is the equivalent traffic load of 312 voice calls. Obviously, a few HDTV channels will require as much, if not more, bandwidth than many voice calls.

Fast packet communications, as mentioned above, are a recent communications method that has evolved to integrate these various types of traffic loads into a single network, allowing for tremendous economies of scale. ATM communications is an International Telecommunication Union (ITU) fast packet communications standard that uses a fixed length packet of 53 bytes, termed a cell. The fixed length of the packets allows simple algorithms to be used for the switching and buffering functions. These simpler algorithms are suitable for hardware implementation, rather than software, which allows much higher bit rates. The other major change from earlier packet communication standards, such as X.25 or HDLC, is that ATM moves the error detection and correction on the data being transferred to the ends of the connection rather than on each of the links connecting intermediate nodes. This also results in the ATM switch being much simpler to implement due to the absence of the error detection and correction components except for the cell header which has

a much lower probability of an uncorrected error than the larger user data portion of the cell due to the HEC protecting the header and the smaller number of bits in the header. As well, ATM will simply drop the cell if an uncorrectable error is detected in the header, leaving it to the end nodes to retransmit or reconstruct the missing data.



a) User Network Interface (UNI) ATM Cell Structure

b) Network Node Interface (NNI) ATM Cell Structure

GFC - Generic Flow Control
VCI - Virtual Channel Identifier
CLP - Cell Loss Priority
VPI - Virtual Path Identifier
PT - Payload Type
HEC - Header Error Control

Figure 1.1 - ATM Cell Format

Each cell has five bytes of overhead, shown in figure 1.1, used to indicate the destination, cell type, cell priority, and a cyclic redundancy check (CRC) byte that protects the five byte header alone from errors. The two header types shown in figure 1.1 illustrate the flow control function at the network edge. The User Network Interface (UNI) form, with the generic flow control field, is for cells entering and exiting the network, while the Network Node Interface (NNI) form, with the extended Virtual Path Identifier (VPI) field, is for cells traveling within the network. The use of the generic flow control field will not be discussed

further, except to note that this field is only present on cells entering and exiting the network. This shows that flow control is not done within the network, but the source(s) transmitting into the network are subject to flow control if it is required.

After the overhead of the header is removed, 48 bytes are left per cell for user data, resulting in an overhead penalty of 9.4 % at a minimum. Notice that the user data is not protected against errors by the CRC byte, which only protects the header. ATM adaptation layers (AALs) have been defined that will protect the user data against bit errors and cell loss at the expense of adding more overhead which reduces the data payload to lower than 48 bytes per cell. These AALs will not be discussed in this report, which concentrates on the actual switching of the fixed length packets or cells, using ATM as a concrete example of fast packet communications. All of the switch architectures presented here are described in terms of ATM switching, but the results are equally valid for any fast fixed length packet communications method.

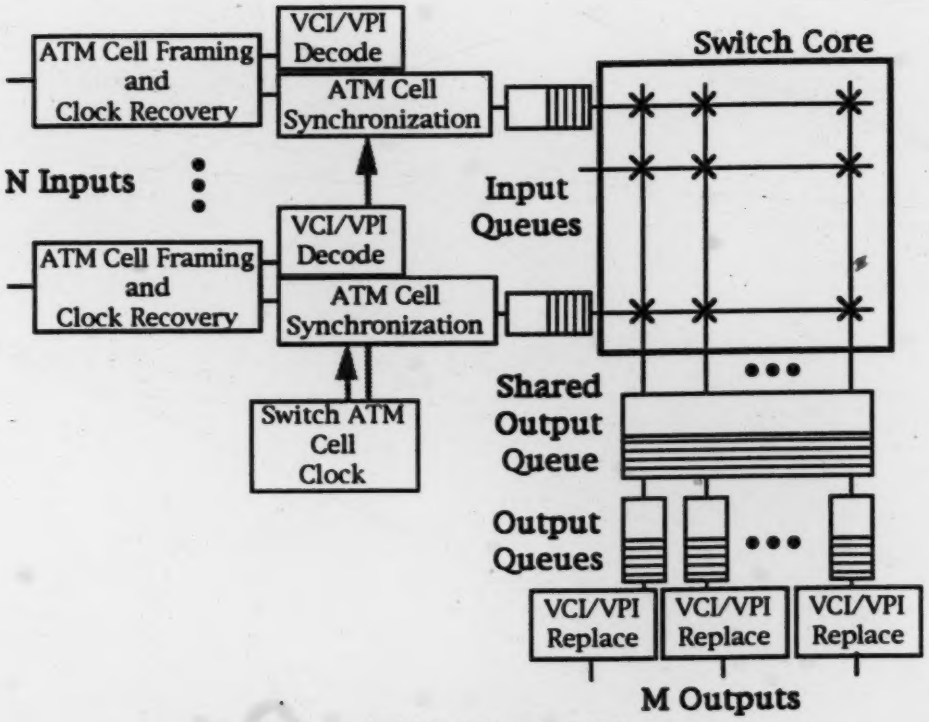


Figure 1.2 - ATM Switch Block Diagram

A logical block diagram of a complete fast packet switch architecture is shown in figure 1.2. A functioning ATM switch may have only some of the blocks shown, depending on the architecture of the switch under consideration. The following functional blocks are shown :

ATM Cell Framing - indicates the ATM cell boundaries of the cells arriving at an input to the switch. This involves framing on the CRC byte in the five byte header for the repeated cells [Dod93][dPr91].

VPI/VCI Decode - decodes the virtual path and circuit identification fields (VPI/VCI) of an arriving cell into the desired output. The VPI and VCI are used to index an associative memory cache for each input, returning the desired output. When a connection is established, the VCI/VPI and desired output entries are added to the associative memory cache for this input.

ATM Cell Synchronization - ensures all the arriving ATM cells are

synchronized at the input to the switch. Cells are delayed at each input in separate small FIFO memories (holding one cell maximum) and are clocked out in unison into the switch core.

Input Queuing - buffers the arriving ATM cells until they can enter the switch core. This memory is in addition to the small ATM Cell Synchronization queue discussed above.

Switch Core - routes the ATM cells from the inputs or input queues to the shared output queues, output queues, or outputs depending on the switch architecture.

Shared Output Queuing - a pool of buffer space shared among all the outputs used to queue the ATM cells before they are sent out the output.

Output Queuing - buffers ATM cells destined for this output until they can be sent out the output.

VCI/VPI Replacement - replaces the VCI/VPI of the ATM cell with the proper VCI/VPI for the output link. Again, this would use an associative memory (in practice, probably the same associative memory) and the VCI/VPI of the incoming cell to provide the VCI/VPI for the output link.

This report concentrates on the switch core and queuing techniques used in developing a fast packet switch architecture. The other components are required for a fully functional fast packet switch, but the design of these components is reasonably well established. For example, the VCI/VPI replacement and desired output decoding operates at the input cell rate and can be implemented using a simple electronic associative memory cache with a 40 nsec access time (for an OC-192 input bit rate of 9.95 Gbps). This is easily done using inexpensive CMOS components.

On the other hand, the memory requirements to buffer up to N-1 of N incoming packets that are destined for a single output are decidedly non-trivial

at gigabit per second data rates. Since any sort of digital logic is very expensive at these bit rates, a large electronic buffer is not economically practical. Lengths of fiber optic strands can be used as the delay lines forming a FIFO (first-in first-out) buffer. An ATM cell at the OC-192 data rate requires a fiber (refractive index of 1.45) length of approximately 8.81 meters to delay one cell interval, so the lengths of fiber required are practical.

ATM switching will be used throughout for examples of fast packet switching schemes. It is assumed that any other fast packet switching schemes will share ATM's properties of : small, fixed length packets; switched (not broadcast) medium; connection oriented (packets all follow the same path through the network); packets arrive in order; and packets may be dropped.

Once again, this report will concentrate on the fast packet switch architectures switching the individual packets themselves. Considerations such as higher level protocols (i.e. TCP/IP, X.25) being transported over the fast packet physical layer, or flow control concerns (i.e. leaky bucket, back pressure schemes) will be ignored. As well, this report will not attempt to give anything other than a very brief overview of the ATM standard. There are several good introductions to ATM already, such as [dPr91][New92][McQ91].

Various switch architectures will be compared to one another based on packet loss probability (PLP) and average and maximum latency. Packet loss probability is simply the probability of a packet arriving at the switch and not leaving on its desired output. The latency is just the delay a packet experiences within the switch from when it arrives to when it leaves the switch and is usually expressed in terms of 'time slots', which is the duration of the fixed length packet at the input bit rate. Theoretical results are presented for the simpler switch architectures, and simulated results are shown for all switch architectures.

The components that may be used to construct an optoelectronic fast

packet switch core are described next, using the terminology scheme from [RIM88], with experimental measurements given where available.

Ideal Switch Architectures

An ideal packet switch will route all the arriving packets to their desired outputs with zero latency (the delay a packet experiences within the switch) even under a full load. The ideal switch will use as small a number of switching elements as possible to increase reliability and decrease cost. These ideal switching elements will be able to change state in zero time, have no bandwidth limitations, have zero crosstalk between the switch elements, and have high isolation between the on and off states of the switch elements. The ideal switch will also use as simple a control algorithm as possible so that it can be controlled in real time, and will treat all of the inputs equally. As a final consideration, an ideal switch architecture will be scalable from a small number of inputs/outputs (i.e. 10 or less) to a very large number (i.e. 1000, 10000, or more), as well as from low bit rates to very high bit rates (multi-Gbps).

Unfortunately, it is impossible to build a practical switch that meets all of these conditions. This can easily be shown by assuming that two packets arrive at different inputs in the same time slot destined for the same output. One of the packets can be delivered to its desired output, while the other packet is either delayed or dropped. Neither delaying or dropping of a packet should happen in an ideal switch. The next two chapters will discuss the trade-offs made in various switch architectures, starting from pure switch classifications to a few of the many switch architectures that have been presented in research papers.

This research project concentrates on high speed (OC-12 bit rates or greater per input) packet switching architectures with fewer than ten inputs and outputs. The design should be expandable both to higher bit rates (minimum of the OC-192 bit rate per input) and more inputs and outputs (at least sixteen).

Switch Buffering Classification

Buffering is required when more than one packet arrives simultaneously at the inputs to the switch, destined for the same output. One of the packets can be routed to its desired destination, while the other packet(s) must be delayed in buffer(s) until the output is free. If no buffer is available, then the other packets can only be dropped from the switch and lost. This leads to an unacceptably high packet loss probability under all but the very lightest of loads as will be discussed later in the section on crossbar switches. Given that buffers are necessary, the question becomes what size of buffer should be used and how should it be implemented. A buffer is simply a memory that will store one or more packets until one of these packets can be read from the buffer. The buffer can be implemented as a random access memory (RAM) from which any packet can be read out at any time, or as a first-in first-out (FIFO) memory from which the packets must be read out in the order in which they were written.

The RAM buffer has the advantage that a high priority packet can share the same buffer as low priority packets, yet still be read out from the buffer without waiting for the low priority packets to be read first. This allows less buffer space to be used and still maintains an acceptable packet loss probability [Kar87]. The disadvantage of the RAM buffer is the added complexity of having to be able to read from and write to more than one location in the buffer.

The FIFO buffer is the simpler to implement because all of the packets are read from the head of the buffer and written to the tail of the buffer, but multiple FIFO buffers are required to implement priority schemes. Since each FIFO buffer has to be sized for worst case packet arrivals, and the memory can not be shared between the FIFO buffers, more buffering space is required to meet the desired packet loss probability than for the RAM buffer [Kar87].

A feature used in classifying switches is the location of the buffers for the

arriving packets. There are four buffering methods normally used in packet switches : buffering at the inputs; at the outputs; shared among the outputs; and along the connection route. The first three of these are shown in a logical block diagram of a switch architecture in figure 2.1.

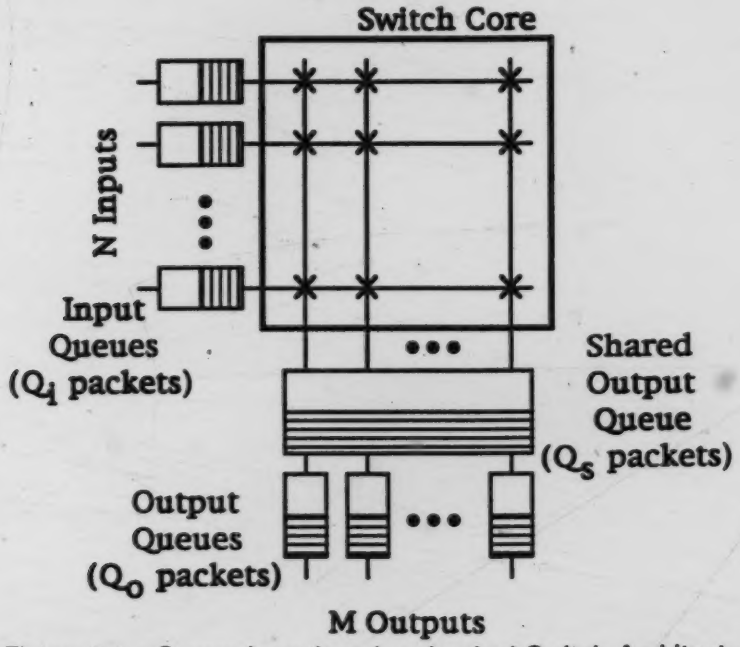


Figure 2.1 - Queue Locations in a Logical Switch Architecture

Input Buffering

With all the buffering located at the inputs to the switch, a packet is delayed at each input until its desired output is free and the packet can be routed to that output. When FIFOs are used as the buffers, a phenomenon called 'head-of-line blocking' occurs which limits the maximum throughput. Head-of-line blocking occurs when the packet about to be read from an input FIFO queue can't be routed because its desired output is busy, but the next packet in line could be routed. Unfortunately, because the buffer is a FIFO queue, the next packet can't be read. This problem leads to extra delay in the switch and a lower throughput under high loads when packet collisions at an

output are more likely to occur. A 'square' switch (having the same number of inputs and outputs) is limited by head-of-line blocking to a throughput of approximately 0.586 [Kar87] with a large number of inputs (for every packet sent into the switch, on average 0.586 of a packet exits the switch). Even with only four inputs, the maximum throughput is limited to 0.655.

The problem of head-of-line blocking can be alleviated by using a RAM for the buffers, which adds complexity and cost, or by using multiple FIFOs to group the packets arriving at an input according to their desired output. Unfortunately, the multiple FIFO buffers cannot be shared easily, which again leads to an increase in the amount of buffering that is required to meet the desired packet loss probability. The advantage of input buffering is its simplicity and that it requires that the switch core only deliver one packet at most to any output in a single time slot. As well, each buffer must store at most one packet and deliver one packet to the switch core in a single time slot.

Simple Output Buffering

When all the buffering is located at each individual output from the switch, an arriving packet is immediately routed through the switch to its desired output. Because each packet is sent to its desired output immediately, there is no head-of-line blocking effect, although the switch core must be capable of sending (in the worst case) a packet from every input to one output buffer in a single time slot. The buffer at the output must be capable of accepting this many packets, as well as sending out one packet to the output in the same time slot. Upon arrival at the desired output, the packet is placed in a buffer and waits its turn to be sent out the output. Since all of the packets in the buffer are for the same output, the latency performance of this switch is optimal (not ideal, as some delay is introduced). With a simple FIFO memory per output, packet priority schemes cannot be implemented, although they can be with a RAM

buffer or with multiple FIFO buffers. Again, multiple FIFOs require more buffer space to meet the desired packet loss probability while the RAM buffer requires a much more complex access controller to determine which packet to read from the buffer next, based on the priority scheme.

Shared Output Buffering

In this case, the buffering again occurs at the outputs of the switch but the buffer space is shared among all of the outputs. Less buffer space is required for this buffering method because packets destined for one specific output can borrow buffer space from any or all of the other outputs if required. Obviously, a single FIFO memory is not appropriate here because a packet should be read into each output that has a packet waiting, in a single time slot. A RAM buffer, or multiple FIFO buffers (at least one per output) are required. Since the main advantage of the shared output buffering scheme is reducing the total amount of buffering space required to achieve a desired packet loss probability, multiple FIFO buffers are inappropriate (and are logically identical to the simple output buffering described above). The larger the number of outputs, the more savings sharing the output buffers make simply because there are more outputs to share buffer space among. The RAM buffer shared among the outputs requires a more complex access controller to track where the next packet for an output is stored. Again, the switch core has to be able to deliver a packet from every input to the shared output buffer in one time slot in the worst case. In addition, the shared output buffer must be able to receive that many packets and deliver a packet to each of the outputs in the same time slot. This buffering scheme also provides optimal latency, a characteristic of output buffering in general, because a packet only waits for those packets destined for the same output.

The switch core shown in figure 2.1 above is not strictly necessary, depending on the implementation of the shared RAM output buffer. The switch

matrix is valuable in some implementations, such as multi-port RAMs, because it ensures that packets destined for the same output (or group of outputs) enter the shared buffer from the same entry port. This may help reduce the complexity of the controller in some cases.

Route Buffering

This buffering method has the buffers dispersed throughout the switch along the routes (in between the switching elements) used in the connection. This type of buffering is only appropriate to switches that use a number of small switching elements internally and have the possibility of blocking in the middle of the switch. When a packet arrives at a switching element that is already in use or set to the wrong state for this packet, the packet is blocked and cannot proceed further. Either the switch has to go through a setup phase to determine which packets will be routed through the switch [Hui87] and buffer the unroutable packets at the input, or the packets will be routed partway through the switch and buffered at the switching element that blocks the packet [Tsu94]. Then, in the next time slot, the packet proceeds further through the switch, until it reaches its desired output or blocks on another switching element. Upon close examination of one of the small switching elements, it appears as a small complete switch itself. Therefore, this type of buffering reduces to one of the three earlier buffer types, and will not be discussed further.

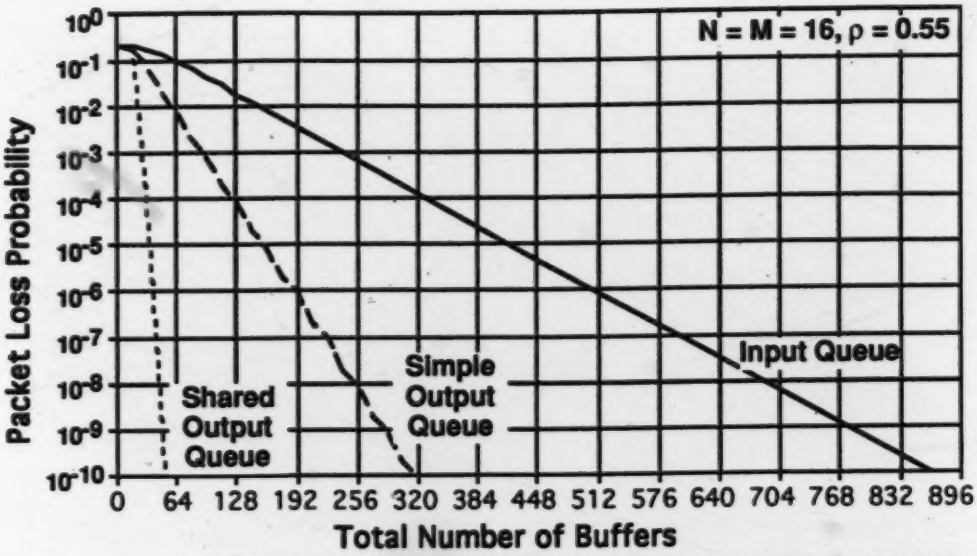


Figure 2.2 - Optimal PLP and Latency for Various Queuing Methods [Mel92]

Figure 2.2 shows the typical performance of input, output, and shared output buffering schemes [Mel92]. These calculations are based on the simplifying assumption that there is no dependence between arrival probabilities on each input and so result in an upper bound on the expected performance for the shared buffer case. This assumption ignores the negative correlation of the inputs which follows from a large number of packets arriving for one output means fewer packets can arrive for the other outputs. This approximation is not strictly valid and tends to overestimate the buffer space required by as much as 30% for smaller switches using shared output buffering [Eck88]. Shown is the total number of buffer spaces required in the switch versus the average packet loss probability (PLP) performance of the switch. As figure 2.2 shows, the shared output buffering method requires the fewest buffer spaces (each holding a single packet) to achieve the desired packet loss probability and has optimal latency characteristics. Shared output buffering requires a complicated buffer controller and a buffer able to accommodate up to 2N packet accesses in one time slot. The input queuing method has the worst

performance (both in terms of buffer size and latency), but requires the least control bandwidth on the internal switch connections and is the simplest to implement. The input queues are only accessed twice per time slot in the worst case. The simple output buffering method also has optimal latency but its packet loss performance is somewhere in the middle. Simple output buffering requires the buffer to be accessed up to N+1 times per time slot, although the buffer controller is simpler than that of shared output buffering.

Shared output buffering offers the best performance of these pure buffering schemes, although hybrid buffering schemes (a combination of two or more of the buffering schemes described above in a single switch) are often used in real switch architectures [Che91b][Pat93]. Unfortunately, the method the switch uses to connect the inputs and outputs will restrict the choice of buffering methods to be used. For example, a switch core that only allows one packet to be delivered to an output in a single time slot will force input queuing to be used.

Switch Core Classification

Since the packets arriving at an input during any time slot may be destined for any output, the switch core must allow any input to be connected to any output in some fashion (except in shared output buffering). There are several methods commonly used to connect the inputs and outputs which are described in more detail below.

For comparison purposes, a switch size of four inputs and outputs and the OC-24 bit rate of 1.244 Gbps per input is assumed for the discussion prototype, and a moderate size of sixteen inputs and outputs and the OC-192 bit rate of 9.96 Gbps per input to demonstrate scalability. These values are used to calculate the required speed of the various switch components needed in order to compare the feasibility of the various switch architectures discussed.

Time Division Multiplexing - Common Bus

In this type of a switch core all the inputs and outputs attach to a common bus and share the bandwidth of that bus as shown in the figure below. The inputs are time division multiplexed (TDM) onto the common bus. This means that each input gets control of the common bus for 1/Nth of a time slot, during which time the controlling input can send a single packet onto the common bus. Each of the inputs and outputs must attach to the bus at the high bit rate, even though each input and output normally only uses a fraction of that bit rate. For high bit rates or large numbers of inputs and outputs, the bit rate of the common bus must be very high, or the common bus will become a bottleneck in the switch.

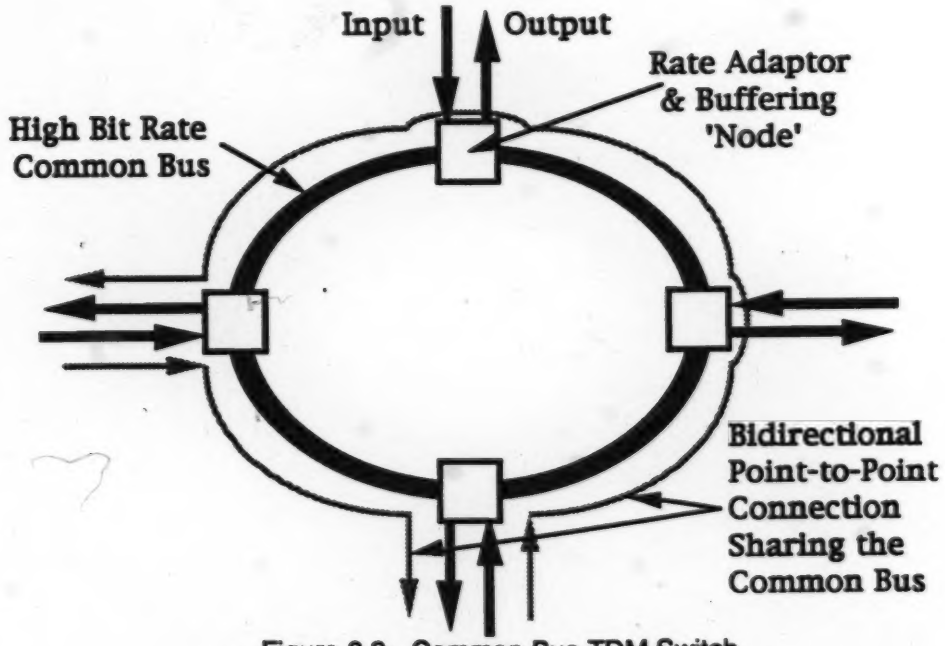


Figure 2.3 - Common Bus TDM Switch

This switch has the advantage of a very simple connection scheme between the inputs and outputs where each input has control of the common bus for 1/Nth of an input time slot and every output receives only the packets

addressed to it. In this manner, during a single time slot every input has the chance to send one packet to any output. Since any input can be connected to any output this switch type is termed 'strictly non blocking'. Obviously, up to N packets can arrive at a single output in every time slot. These packets have to be buffered at the individual outputs and sent out one per time slot to the output line. To have zero packet loss probability requires that every output have an infinite buffer. Choosing an acceptable packet loss probability (normally 10^{-9} which is approximately equal to the bit error rate of the physical media times the number of bits per packet header) and knowing the probability of arriving packets destined for a specific output (application dependent), allows the required size of the buffer to be determined either through calculation or simulation. This buffering scheme provides the best possible latency characteristics due to the output buffering scheme used.

This switch type is very good for constant bit rate, bi-directional point-to-point traffic (such as uncompressed voice) because the two directions can share the common bus (as indicated by the gray lines in figure 2.3) without requiring another complete set of switching hardware for the reverse direction. Unfortunately, if point-to-point connections are not used (i.e. a broadcast connection), or if the traffic is not similar in both directions, this advantage is lost.

The disadvantages of this switch type are that every input and output must be capable of sending and receiving at the high bit rate of the common bus, even though each input and output individually normally uses only a fraction of that bandwidth. Another problem with this switch is its limited expandability because of the bottleneck of the common bus. Given the switch sizes mentioned above, the small switch of four inputs and outputs at OC-24 bit rates requires a common bus with a bit rate of 4.8 Gbps. This is currently expensive, but is feasible and will only get less expensive as time passes. For

the larger switch, with 16 inputs and outputs at OC-192 bit rates, the common bus must have a bit rate of 1.5 Tbps. This is definitely not feasible in the near future.

There are several examples of this type of switch, including the Forerunner ASX-100 ATM switch that is commercially available from FORE Systems [FOR93]. The ASX-100 switch uses a 2.4 Gbps bus with four input and output OC-12 connections. Each of these four ports can then be shared further between four OC-3 connections, resulting in a 16x16 ATM switch. The Athena switch [dPr87] was one of the earliest ATM switches to use a TDM common bus for switching cells.

Time Division Multiplexing - Shared Memory

This switch core uses a shared memory between the inputs and outputs. The inputs generally feed into the shared memory through a common bus with one write allowed for every input during a time slot. The outputs are read from the memory in the same fashion. The memory can be implemented as multiple FIFO buffers or as a RAM, although the RAM is preferred as it will allow a higher throughput by avoiding the head-of-line blocking phenomenon described earlier. The required access time for the RAM will be N input writes and M output reads per time slot to assure that no packets are lost. If this requires a RAM with too fast an access time, individual RAM's can be assigned to each output to increase the allowable access time to N input writes and 1 output read per RAM per time slot, although this does cause an increase in the size of RAM required to maintain the same packet loss probability. Another possibility, although more drastic, is to only allow a fraction of N writes to proceed in a single time slot [Yeh87] and drop any packets in excess of this. A RAM can be assigned to groups of the outputs to attain some of the benefits of shared output buffering while still maintaining a reasonable required access time for the RAM.

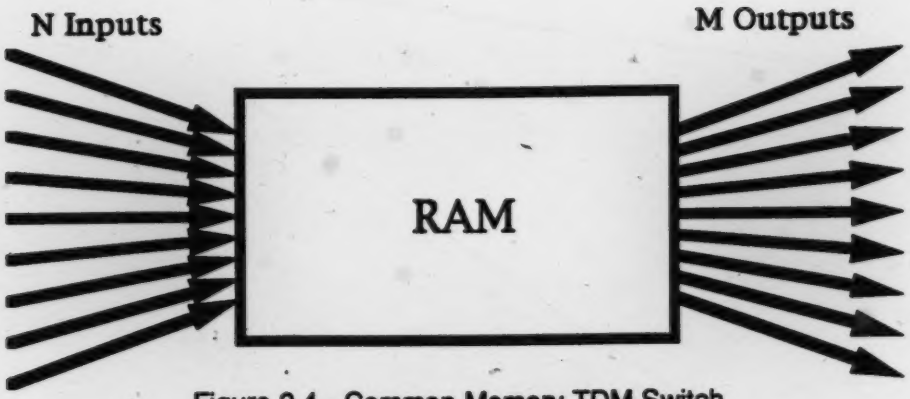


Figure 2.4 - Common Memory TDM Switch

This type of switch also has a very simple connection scheme that consists of writing the arriving packets into the RAM and reading the desired packets out of the RAM for transmission to the outputs. Packets are only lost when there is not enough space in the RAM to hold another packet, or when too many packets arrive in the same time slot destined for the same output and the RAM cannot be accessed quickly enough to store all of the packets. A RAM is needed for each direction of a bi-directional connection in all cases, unlike the common bus TDM switch.

The greatest disadvantage of this switch type is that the memory has to have a fast access time and is very expensive. For our two switching examples, the demonstration switch would require an access time of 42 nsec per cell as calculated using equation 2.1 below, while the larger, faster switch would require an access time of 1.3 nsec per cell assuming fully shared output buffering with no possible packet loss at the input to the RAM. If simple output buffering is assumed, again with no possible packet loss at the input to the RAM, the smaller switch requires an access time of 68 nsec per cell, while the larger switch requires an access time of 2.5 nsec per cell. To achieve a packet loss probability of 10^{-9} with a uniform traffic load of 0.55 packets arriving at each input in a single time slot (destinations uniformly distributed over all outputs),

the fully shared output buffering case requires a minimum of 4 buffers per output, while the simple output buffering switch requires a minimum of 18 buffers per output (see figure 2.2).

$$t_{access}\ \text{sec} = \frac{(\#inputs + \#outputs\ accesses) \cdot (424\ bits / cell)}{bit\_rate\ bits / \text{sec}} \qquad \text{Eq. 2.1}$$

Examples of this type of switch are the Roxanne and Coprin switch architectures [dPr91], as well as the Prelude [Cou87].

Space Division Multiplexing - Signal Routing

Space division multiplexing (SDM) switches use different routes through the switch core, rather than different time slots on the same route. Since the connection between an input and output is not shared, the complete bandwidth of a connection can be dedicated to that input/output pair. Routing SDM switches actually route the signal from the input through the switch to the output, ideally preserving the signal power through each stage of the connection so no amplification or regeneration of the signal is required. Some of these switch architectures are 'blocking', such as the Banyan architecture in figure 2.5a, which shows a connection that cannot be completed between a free input and a free output due to another connection that is already established. Other architectures are termed 'strictly non blocking', such as the Clos or the Batcher-Banyan network topology, which is shown in figure 2.5b, where a connection between a free input and output can always be made, while other architectures, such as the Benes network, are 'rearrangeably non blocking' [Tob90], which always allows a connection to be made between a free input and output although some of the already established connections may have to be rearranged. As one would expect, the blocking architectures use the fewest switching elements, the strictly non blocking architectures use the most switching elements, with the rearrangeably non blocking architectures in

between.

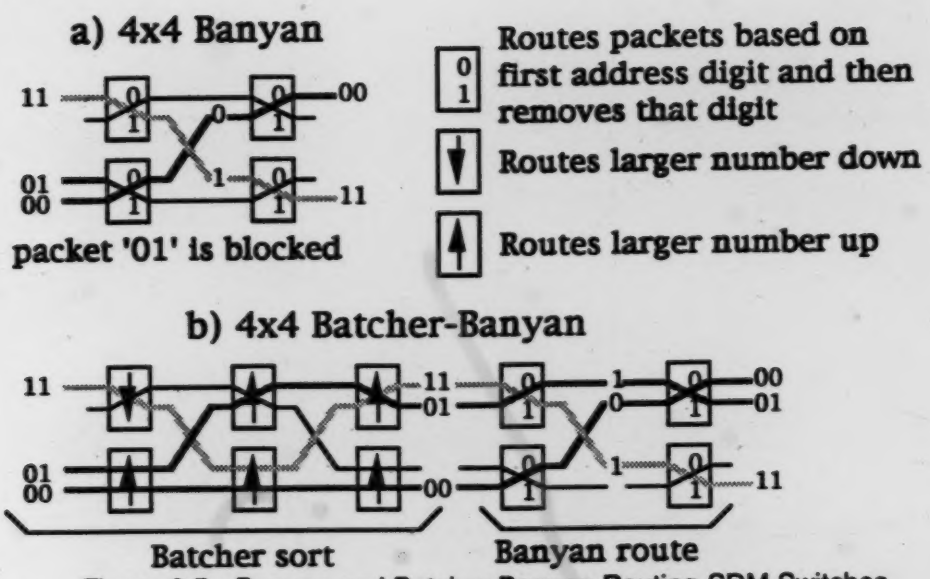


Figure 2.5 - Banyan and Batcher-Banyan Routing SDM Switches

The advantage of SDM switches in general over TDM switches is that the entire bandwidth of a connection is dedicated entirely to one input and output combination. This means that the addition of more inputs and outputs does not reduce the bandwidth available to the existing input and outputs, and the switch is expandable to larger numbers of inputs and outputs. In addition, the control of the routing SDM switch can be done either centrally (one controller controlling all of the small routing elements), or in a distributed manner (every small routing element decides whether to pass or block its inputs on its own). The distributed control allows higher bit rates as each routing decision is much simpler, but the switch architecture requires more care to design and uses more routing elements than with central control. In addition, for high speed optical signals that use distributed control, the switching element must either be optically controlled by the incoming signals (as in an optical non-linear switch), or a portion of the optical signal must be tapped off and used for electrical control (for example, with a Mach-Zehnder switch). These small portions of the

signal being tapped off at each switching element will eventually require replacement through amplification, especially when many stages of switching elements are required in series, as in a large dimension switch.

The advantage of the routing SDM switch is cost. It only requires 0.5 N $\log_2(N)$, two input by two output switching elements to implement a Banyan switch fabric [Tob90] with N inputs and N outputs that requires either a central control to manage internal collisions, or switching elements capable of buffering arriving packets. A distributed control Batcher-Banyan non-blocking switch requires 0.25 N $((\log_2(N))^2 + 3 \log_2(N))$ switching elements, still less than the N^2 switching elements required by a simple crossbar design which will be described below. Notice that each of these switching elements only passes the bit rate of an individual input/output connection, unlike the TDM switches.

If optical input and output switches are used in a switch core, the data path through that switch is 'optically transparent', meaning that the optical signal is not converted to an electrical signal anywhere inside the switch. This is important because increasing the bit rate of the switch core means only changing components at the periphery of the switch, rather than having to replace the entire switch core as well.

One of the problems associated with this type of switching is the availability of the small switching elements. Electrical switching elements can be integrated in an ASIC which allows the simple implementation of the buffer and control, but only at relatively low bit rates compared to optical signals. Small optical switching elements are commercially available now that have switching bandwidths of 18 GHz [UTP94] and far higher signal bandwidths since they are optically transparent. These optical switching elements have relatively large insertion losses of 5 dB, low isolation between the on and off states of 20 dB, and are extremely large and expensive. These devices require

amplification to compensate for the insertion losses, which adds even more cost and noise and decreases reliability of the overall system. The low isolation between states limits the number of switches that can be cascaded together to form a larger switch without requiring the signal to be regenerated. In addition, the buffering of optical signals is problematic as will be discussed later. Research work is ongoing with improvements available both in component integration [Jan94] and switching characteristics [Jin92], but due to the nature of optical switching devices, large numbers of these switching elements will not be integrated together into one wafer in the near term [Mid93b]. As well, these optical switching elements currently require electrical control which becomes a limiting factor in the maximum packet rate [Bit91]. Another problem is that each of the switching elements and amplifiers will degrade the signal to noise ratio of the signal as it passes, through signal losses, added noise, and crosstalk. Since each optical switching element only selects two paths the signal passes through many stages in a large switch, requiring regeneration of the digital signal in some manner. Currently such regeneration (consisting of amplification, thresholding, and retiming of the digital signal) is only feasible in electronics, which will also limit the maximum bit rate. Recent research has produced an optical regeneration system [Wei94], although this is far from a commercially available product yet.

Broadcast connections are very complex to implement in these switch architectures, with the simplest method being to actually duplicate the packet as it arrives and send these duplicated packets through the switch individually [Gia91]. These extra packets add to the congestion of the switch, increasing the packet loss probability and latency, as well as requiring extra components to actually duplicate the packets.

Examples of this type of switch with distributed control are the Batcher-

Banyan switch fabric [Gia91], while centralized control switches generally use the Banyan architecture to minimize the number of crosspoints [Hua84]. Electrical implementations of the Banyan architecture using VLSI circuits [Vai88] have also been demonstrated. Since the centralized electronic controller becomes a bottleneck when very high bit rates or large numbers of inputs and outputs are used, photonic switches of this type almost always use the distributed control method. An interesting example [Tac94], although not suitable to fast packet switching, uses a three stage Clos network and robotic arms to manually connect optical fibers to form a 512x512 switch with a connection setup time measured in minutes.

Space Division Multiplexing - Signal Distribution

Signal-distributing switch cores split the incoming signal from each input to all of the outputs. The signal from a specific input is then either passed or blocked at each of the outputs to achieve the desired input to output connections, ignoring any queues involved. Since each output requires a connection from every input, this switch type requires N^2 switching elements that either pass the signal (on state) or block the signal (off state). This type of architecture is strictly non blocking since every input is connected to every output.

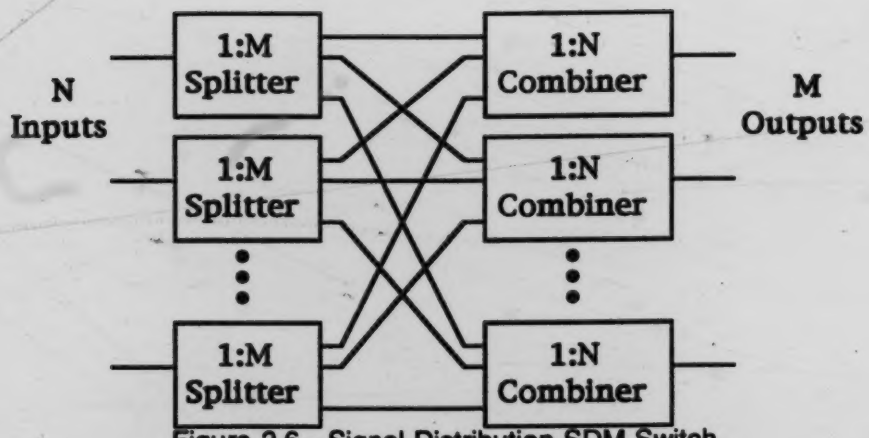


Figure 2.6 - Signal Distribution SDM Switch

Again, the signal distribution SDM switch has the entire bandwidth of a connection dedicated to each input/output pair, allowing the switch to be easily expanded as in the routing space division case. This switch type is usually controlled from one central controller which sets all of the switching elements. Central control allows for the simple implementation of broadcast connection by having more than one output connected to a single input.

The largest problem with this switch type is the large losses associated with splitting the signal between all of the outputs. For high speed signals with simple power splitting, half the power in the signal is lost for every two way split. For a four output switch, every output has one quarter of the arriving power, while for a sixteen output switch, only one-sixteenth of the power is delivered to each output. These losses must be compensated with amplification, either before or after the signal is split. In addition, the central controller currently must be implemented electronically which limits the maximum packet rate, especially as a large number of switching elements must have their state set (the product of the number of inputs and outputs).

Examples of this type of switch are the Triquint GaAs crossbar switch chip [TRI92] and optoelectronic switch matrices [Lam94][For89].

Wavelength Division Multiplexing - Multihop Switches

Wavelength division multiplexing (WDM) switch cores transmit each input signal on a different wavelength of light. In multihop WDM switches, each input and output of the switch (termed a 'node') is assigned a particular wavelength to transmit on, and a different wavelength to receive on. These wavelengths are fixed during the design of the switch. When a packet arrives at a node, it is broadcast by that node on its particular wavelength. A different node of the switch, which receives on that specific wavelength, receives the packet and determines if the packet was destined for this node or not. If it is, the packet is received and sent out this node's output, otherwise the packet is sent out again on this node's assigned transmitting wavelength. In this fashion, the packet is bounced among the nodes, changing wavelength with each hop, in the switch until it arrives at the desired output node, as seen in figure 2.7. Naturally, each 'hop' between the nodes in the switch adds delay to the packet. For a uniform destination probability, the average in-transit latency will be half of the number of nodes times the latency introduced at each node between reception and transmission. In addition, there will be some latency for the packet when it arrives at a node due to that node already transmitting a packet.

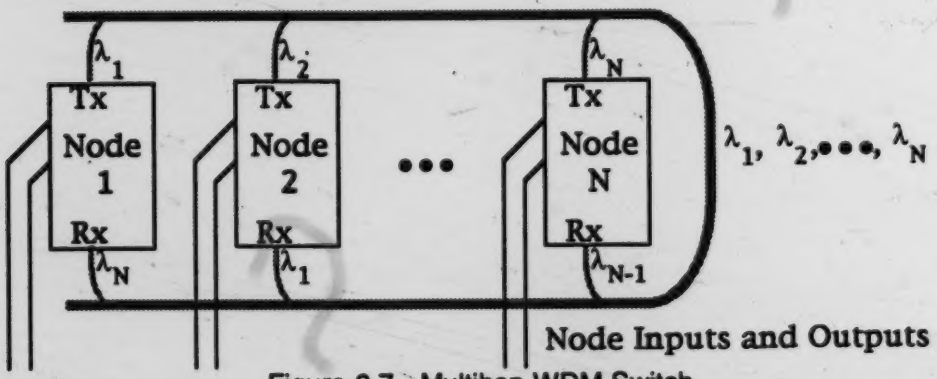


Figure 2.7 - Multihop WDM Switch

This switch architecture is strictly non-blocking, because any free input

will eventually reach any free output. It also allows for very simple broadcast connections by repeating the packet to the next node until it has reached all of the nodes it is destined for. The hops between the nodes act as a small shared output buffer, although not strictly as a RAM or FIFO buffer, due to the differing delay experienced by a packet destined for a specific output depending on which input the packet arrived at. When the switch core operates at the same bit rate as the inputs and outputs, only one packet can be delivered to an output in a single time slot so output buffering is not applicable. Input buffering must be used in addition to the small shared output buffer to lower the switch's packet loss probability if buffering is necessary.

Generally, a single fiber connects the nodes in the switch, although some architectures use two fibers to reduce the delay and increase the reliability [Mid93b]. This single fiber is shared among all the nodes using N different wavelengths, with each node using precise wavelength dependent couplers to extract its receiving wavelength and add its transmitting wavelength with minimal losses. For ideal components, this adding and dropping of wavelengths would have no loss at the other wavelengths. Unfortunately, ideal components do not exist, so excess loss will be present and must be compensated for, either with optical amplification or larger signal powers.

Notice that this switch architecture implies at least one electrical to optical (E/O) conversion when a packet enters the switch, one optical to electrical (O/E) conversion and one E/O conversion at each intermediate node encountered, and at least one final O/E conversion when the packet leaves the switch. Having this many conversions will introduce a significant amount of noise, requiring either signal regeneration or high signal powers to keep the bit error rate low. Another problem is the reliability of the nodes. If a node fails, the switch no longer can send any input to any output because the wavelength

chain has been broken. In addition, the problem of keeping the separate components calibrated to the same wavelength despite environmental changes and aging is not a trivial problem [Mid93b], especially when the different wavelengths are spaced closely together as they will be in a large switch. This leads to problems in expanding the switch to have more inputs and outputs.

One of the first examples of this switch architecture was the ShuffleNet [Aca87]. Other switch architectures of this type are usually generalizations of the ShuffleNet that minimize the number of wavelengths used, such as the one-hop network [Mid93b] which proposes a hierarchical grouping of nodes that allows wavelength reuse.

Wavelength Division Multiplexing - Wavelength Agile Components

This type of switch core also uses different wavelengths to transfer the packets between inputs and outputs. The difference is that this switch architecture uses components that can change their wavelength rather than fixed wavelength components. For most architectures of this type, either the transmitters or receivers are wavelength tunable, while the complementary element uses a fixed wavelength. Some architectures use both wavelength tunable transmitters and receivers which allows for wavelength reuse (i.e. the switch can have fewer wavelengths than inputs or outputs). Figure 2.8 shows an architecture where the both the transmitters and receivers are wavelength tunable [Art88] and the transmitters and receivers are joined with a N to M coupler. The combining of the N transmitters and the splitting to the M receivers will transmit only $1/NM^{th}$ of the signal power from each source to a single receiver because wavelength dependent splitting and combining components cannot be used since the transmitting and receiving wavelengths change. Because of this, signal amplification will be necessary for all but the smallest of switches [DeZ94]. If only tunable transmitters or receivers are used (but not

both), then the power loss due to the splitting would be 1/Nth or 1/Mth respectively.

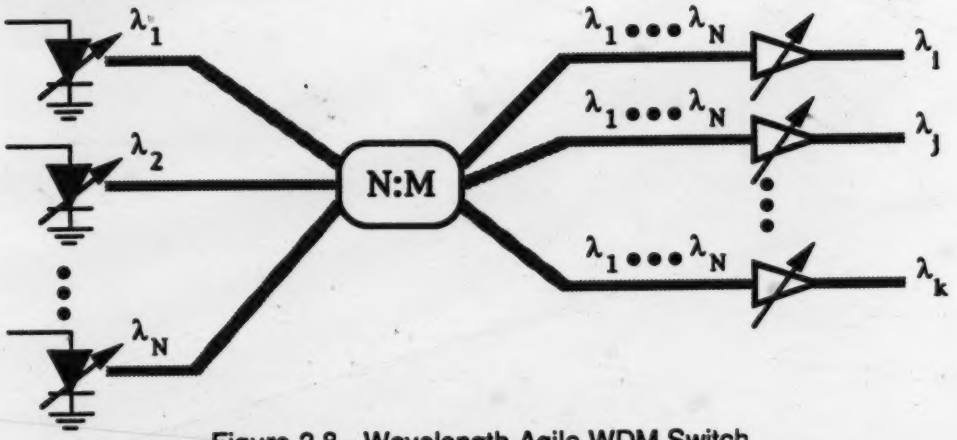


Figure 2.8 - Wavelength Agile WDM Switch

This switch architecture is strictly non blocking and is logically equivalent to a crossbar switch, with the ability to connect any input to any output. There is no buffering in the switch core, unlike the multihop WDM switch. If the switch core is operated at the same bit rate as the inputs and outputs, only one packet can be delivered to an output in a single time slot, leaving input buffering as the only applicable queuing method. As always, input queuing limits the maximum throughput of the switch due to the head-of-line blocking effect.

The biggest problem with the tunable wavelength switch architecture is the availability of the actual components. The components have to have very precise wavelengths, and the ability to change from one precise wavelength to another very quickly. This technology, while promising for the future, is not currently feasible. In addition, there is an implied E/O and O/E conversion at the entrance and exit of the switch, which will limit the maximum possible bit rate in comparison to an optically transparent switch (which keeps the signal in the optical domain throughout the switch).

There are a few examples of this switch type, including the RAINBOW

switch [Don90], the HYPASS switch [Art88], and a 16000 input and output switch [Cis91].

From the comparisons above, it is obvious that the two time division multiplexing schemes are not appropriate for the larger, higher speed switch. The shared bandwidth of the common medium is enough for a small number of inputs, but with a moderate number of inputs or very high input bit rates it becomes a bottleneck. There are methods to alleviate this bottleneck, such as using a parallel bus, but they negate the advantage of fewer components and only move the bottleneck to a higher bandwidth, while adding problems of their own such as electromagnetic interference and crosstalk. Similarly, the access time required for the RAM in the shared memory switch type is not feasible with current technology for all but the smallest switches. Either of the WDM switches could potentially meet both the specifications we have set. However, the WDM technologies are not mature enough currently to be used in developing a switch for the near term.

Either of the space division multiplexing schemes have the capacity to fulfill the specifications for the larger, higher speed switch, although serial electrical communications are not fast enough. Either optical or parallel electrical communication methods must be used. This routing scheme uses fewer components, though they are active components with extremely high bandwidths which currently cost more and have higher insertion losses than passive components. Small switches of this type (four or eight inputs and outputs) have been demonstrated [Gus93][Gra94] using Mach-Zehnder based optical interference switching elements, usually fabricated from lithium niobate technology. A larger switch, with 128 inputs and outputs, has also been partially demonstrated [Bur92] in what was termed a 'hero' experiment [Mid92]. This large switch, when fully completed, would use 8064 two input by two output

optical switches, 784 semiconductor laser amplifiers, and 4288 single mode connections. This technology has many advantages, especially when considering the very high speed switches that will be required in the future. Unfortunately, the components used in this switch type are not mature enough currently [Bit91], and less expensive methods can be used to build a switch that meets near term expectations. This leaves the signal distribution SDM scheme as the basic switch architecture of choice, preferably in a form that can take advantage of shared output queuing to maximize throughput, and minimize latency and buffer requirements as described in the previous section.

Switch Core Technology

The signal distribution scheme uses passive components to split the signal evenly among all of the outputs. Each output then receives the signal from one input and ignores the signals from the rest of the inputs as determined by the controller algorithm. The form of the signals is still undetermined, although there are four possible choices, based on the input to and the output from the switch core being electrical or optical. All-electrical switches are too slow, (although GaAs devices are becoming faster all the time [TRI92]) and have problems with crosstalk, electromagnetic interference, and stray capacitance build-up. As shown in figure 2.9a, there is the additional problem of routing the input, output, and control signals in one plane to the actual switching element. Almost all-optical switches, which use an electronically controlled optical switching element [Nis93] to block or pass the signal, are impractical for the same reasons as the routing space division multiplexing switches described above. They do have the advantage of being optically transparent, as well as only having to route the control signal in one plane, as shown in figure 2.9c, while the input and output optical signals are in other planes.

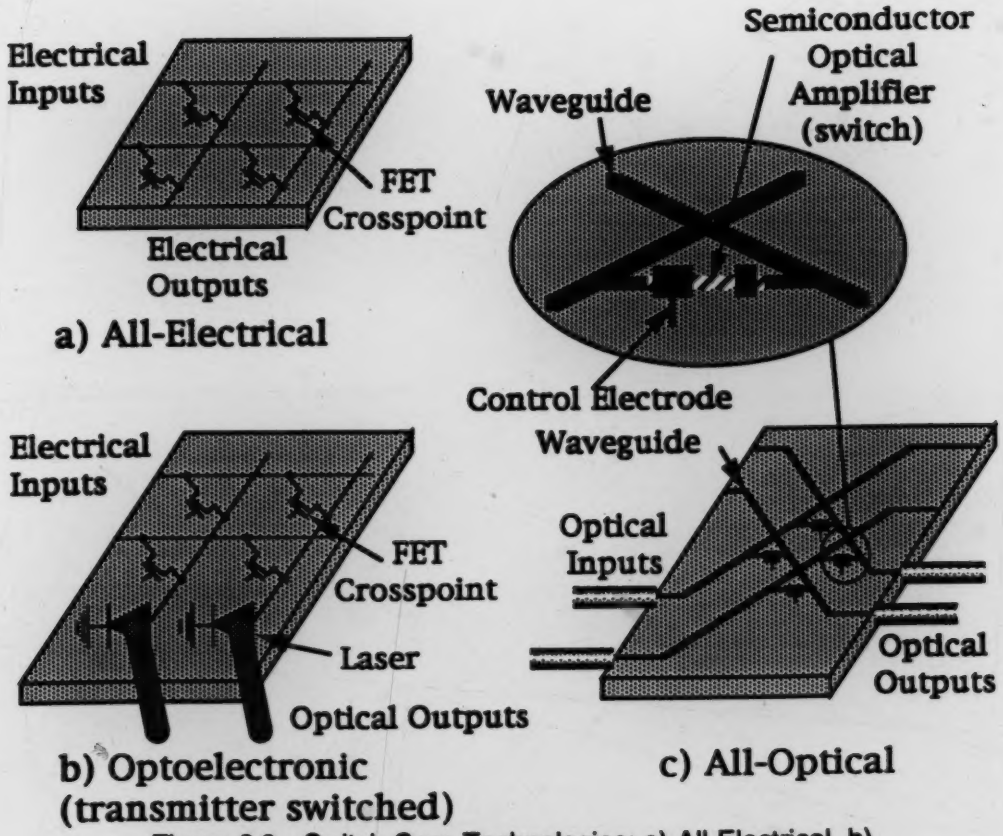


Figure 2.9 - Switch Core Technologies: a) All-Electrical, b) Optoelectronic (transmitter-switched), and c) All-Optical

This leaves optoelectronic signal distribution switching, which delivers the control and one signal in one plane to the switching element as figure 2.9b shows, while the other signal is optical and in a different plane. In the optoelectronic transmitter-switched version, the input signal is split electronically, amplified if necessary, and delivered to optical sources. The electronic control signal then passes one electrical input signal to the optical source, while blocking the signals from the other inputs. The optical output then carries the signal information. For the optoelectronic receiver-switched version, the input signal, if electronic, directly drives an optical source. This optical source is then optically amplified if necessary, and split among all of the

outputs. Each output has a photodetector that detects the optical signal from each input, and the electronic control signals then allow one of the photodetectors to be selected to deliver the electronic output signal, as shown in figure 2.10. In either case, the signal splitting is done passively, so it is very reliable, while the circuitry for the selection of which input to be connected to each output is electronic and can be integrated very easily. Having an optical input or output signal eases the congestion of routing electronic signals to the switching elements, as well as decreasing the crosstalk and electromagnetic interference of the switch.

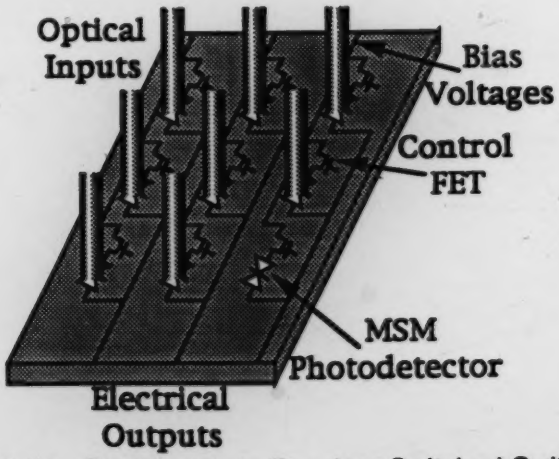


Figure 2.10 - Optoelectronic Receiver-Switched Switch Core

To decide between the two optoelectronic switching versions, consider the operation of an ATM switch. First, the signals arriving at the switch and leaving the switch are high bit rate signals (more than 1 Gbps) that will travel longer than LAN distances (more than 100 meters). It is assumed that the input and output signals are optical to meet these conditions. The arriving cells have the VCI/VPI portion of their headers decoded to decide upon the desired output, and are then switched to it, the VCI/VPI is replaced in the cell header, and the cell is sent out of the switching node. If the cell belongs to a broadcast connection, the cell is switched to all desired outputs, the VCI/VPI is replaced in

the cell header at each output (the new VCI/VPI is not necessarily the same at all outputs), and the cells are sent out. The decoding of the VCI/VPI in the arriving cell must be done electronically, which means that at least some of the arriving optical signal will have to be detected electronically and used to select the route. The replacement of the VCI/VPI header in the outgoing cell must also be done electronically, although here an entirely new signal must be generated for the cell header. To do this, the output signal from the switch fabric must be electrical (converted from optical if necessary), the VCI/VPI must be updated in the header, a new CRC byte for the header calculated, and then the electrical output signal used to drive an optical source to create the optical output signal which exits the switch. Evidently, the signal must be available in electronic form at the output of the switch fabric, though it may be optical at the input to the switch fabric. A small portion of the input optical signal will have to be converted to electrical form. The receiver-switched optoelectronic switch type is much better suited to this use than the transmitter-switched optoelectronic switch type.

The VCI/VPI replacement described above also demonstrates why an all-optical ATM switch is not currently feasible. The optical signal would have to be converted to an electrical signal at some point to replace the VCI/VPI and calculate the new CRC byte for the header. This conversion process negates the higher bit rates possible in an all-optical switch.

As an additional consideration, consider how the two switch types integrate to form a smaller, more compact, more reliable device. To integrate the transmitter-switched optoelectronic switching scheme, N^2 high speed switching elements (such as FETs) must be integrated with N high speed optical sources. On the other hand, the receiver-switched optoelectronic switching scheme requires the integration of N^2 high speed switching elements (such as

FETs) and N^2 high speed photodetectors. High speed photodetectors with very simple structures exist, such as metal-semiconductor-metal (MSM) photodetectors that can be integrated quite easily [Hur91] with GaAs FETs, while a high speed optical source is not a simple structure at all [Pet88][Koh94]. Again, the receiver-switched optoelectronic switching scheme is the more practical of the two.

Based upon these considerations, the desired switch architecture was determined to be a signal distribution space division multiplexed switch core that uses receiver-switched optoelectronic elements and implementing shared output buffering. Switch architectures that are suitable for this are discussed and compared in the next section.

Physical Switch Architectures

In this section, several switch architectures taken from published literature, as well as the reflex switch architecture developed by the author, will be described, and their advantages and disadvantages discussed. The switch architectures described were chosen on the basis of their suitability for the use of fiber delay lines as buffers, photodetectors as crosspoints, and output queuing in some form to minimize the latency and buffer size required. These switches will be compared as to packet loss probability (PLP) performance, latency (both average and maximum), and the number of crosspoints required to implement the switch. The theoretical performance of a few of the simpler switches will be calculated, and simulation results will be presented in most cases.

Monte Carlo simulations have been performed using a uniform load (i.e. a single Bernoulli process) attached to each input with a probability, ρ, of generating a packet in each time slot. Each packet generated also has its destined output chosen uniformly from all of the outputs. This simple source model was chosen because it is the 'common sense' model to be used when many lower bit rate sources are multiplexed together to form a single high bit rate input, such as in statistically multiplexed voice traffic. In addition, this source model is the only model consistently used in other work, and is simple enough to make theoretical analysis of the simpler switch architectures tractable. All of the simulations were performed using a C language program developed specifically for this project. A sample set of simulation input and output files are presented in Appendix A.

The simulation was checked for accuracy by comparing simulation results with the calculated results for a crossbar switch. The simulation results

were also compared to results in the literature for more complicated switch architectures, such as the staggering switch and the reflex switch with staggered buffers.

Unfortunately, evidence has recently been found that the uniform load source model does not apply to traffic that has been multiplexed from various variable bit rate data sources [Lel94] such as the traffic present on a large Ethernet local area network. The uniform load source model obviously applies for constant bit rate traffic (such as voice or uncompressed video) and may or may not apply to compressed video sources. As well, no consensus has been reached yet for a more appropriate source model to be used for switch simulations. For these reasons the uniform load source model was retained for this project.

If correlated sources were used, the packet loss probability results presented here using the uniform source model would underestimate the actual results. This is because the correlated bursts of packets would arrive in a short time frame all destined for the same output. This would either increase the packet loss probability or require more queue space to keep the packet loss probability constant.

All of the switch architectures presented are described in terms of the number of inputs, N, the number of outputs, M, and the number of buffers, P, as shown in figure 3.1. In addition, when a switch architecture uses shared output buffering, the terms inlets (entrances to the switch core from the inputs and the shared memory), outlets (exits from the switch core to the outputs and the shared memory), and the size of the output queue(s), Q_O, will be used. In all of the switches presented here, the inputs and outputs operate at the same bit rate, while the core of the switch is at the same or a higher bit rate. This is termed the 'core speedup', L, where a core speedup of two means the core of

the switch operates at a bit rate twice that of the inputs and outputs. This speedup factor allows the core of the switch to deliver more than one packet to a single output in any one output time slot requiring the switch to use some form of output queuing. Time slots are always defined in terms of the operating bit rate of the inputs and outputs, unless it is specifically stated otherwise.

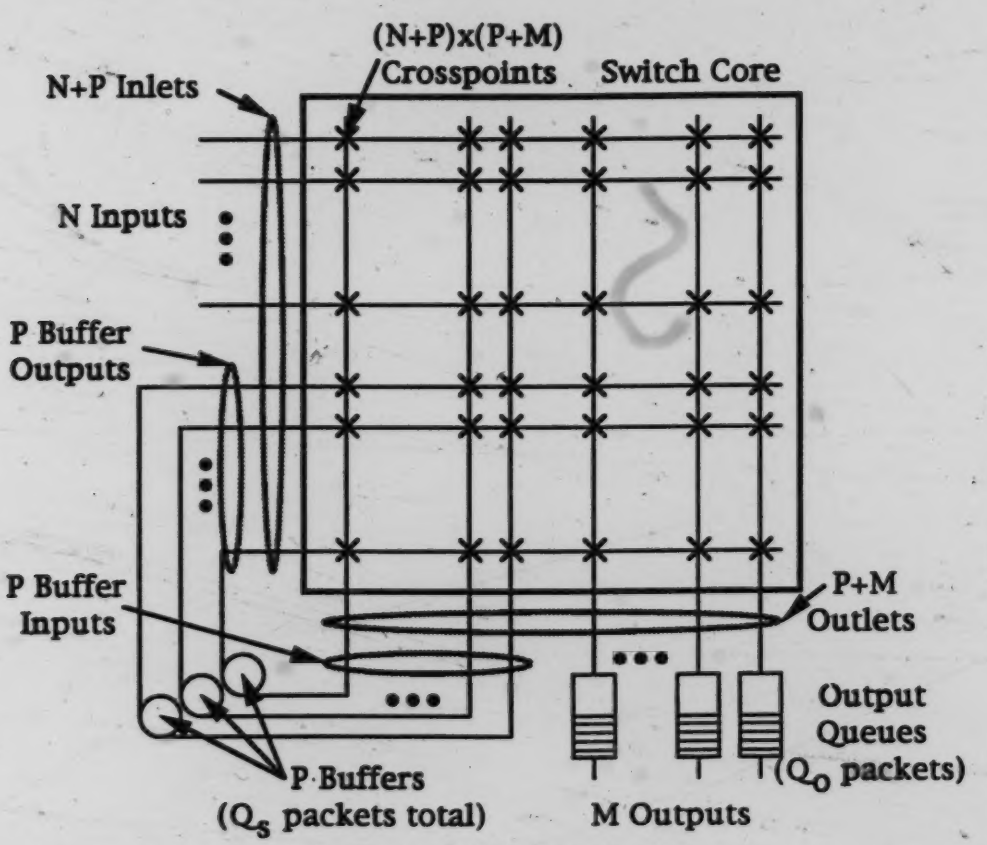


Figure 3.1 - Switch Components

Crossbar Switch

The crossbar switch, drawn in figure 3.2, simply consists of N x M crosspoints that allow any input to be connected to any output. Packets are lost when more than one packet arrives in the same time slot destined for the same output. The core of the switch must operate at the same bit rate as the inputs and outputs as no buffering is used. This switch architecture is strictly non-

blocking, as well as having absolutely no buffering. Obviously, the longest possible delay in this switch is zero (the ideal amount of delay), and all of the packets passed through the switch are kept in order. Unfortunately, the packet loss probability will be very high under all but the very lightest of loads.

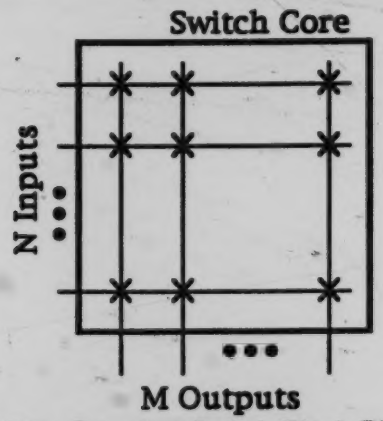


Figure 3.2 - Crossbar Switch Block Diagram

A simple controller, one that always assigns the inputs to their desired outputs in the same order during a time slot, will not treat the inputs fairly. The first input to be checked will always have its packet sent to the desired output (i.e. have a PLP of zero), while the last input to be checked will have the highest packet loss probability. To treat the inputs fairly, the controller must rotate or 'shuffle' through the inputs in each time slot, so that the first input to be checked in one time slot will be the second input in the next time slot, then the third, and so on. This results in the controller algorithm shown in figure 3.3. As it is desirable to treat the inputs fairly, shuffling of the inputs will be assumed, although not explicitly shown, throughout the rest of this document as all the other switch architectures discussed have this same problem and solution.

```
                Crossbar Switch Control Algorithm

for each time slot
      increment the starting input
      if the starting input is greater than the number of inputs then
            set the starting input to input 1
      endif
      for each input (beginning with the starting input)
            if a packet is not arriving at this input then
                  goto next input
            endif
            if the desired output is free then
                  route the packet to its desired output
                  goto next input
            endif
            the packet is blocked so drop it
      next input
next time slot
```

Figure 3.3 - Crossbar Switch Control Algorithm

This switch architecture is not practical for a real switch, but is presented here to be used as a check on the simulation results. Because the architecture is so simple, a closed form expression for the packet loss probability, given in equation 3.1, is easily derived, as presented in Appendix B. Figure 3.4 demonstrates the simulated and theoretical PLP performance of the crossbar switch versus the traffic load offered per input. The lines are the theoretical results, while the symbols indicate the simulated results. The simulation results, shown in Appendix B as well as in figure 3.4, agree very well with the predicted theoretical results, which lends confidence to the simulations performed. Confidence intervals of 95% for the simulation results have been calculated in Appendix B but are not shown in figure 3.4 because the confidence intervals are insignificant. The largest 95% confidence interval for this set of simulations was +/-1.63x10^{-5} from the average packet loss probability.

$$PLP = 1 - \frac{M}{N \cdot \rho} \cdot \left[1 - \left(1 - \frac{\rho}{M} \right)^{N} \right] \qquad \text{Eq. 3.1}$$

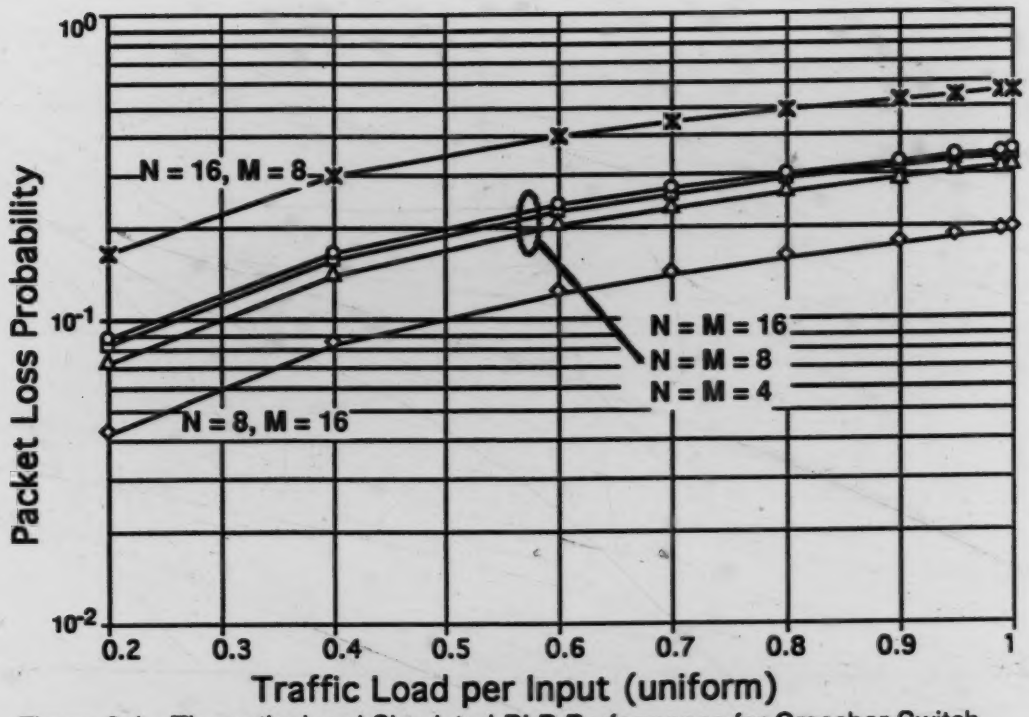


Figure 3.4 - Theoretical and Simulated PLP Performance for Crossbar Switch

For the desired PLP performance of 10^{-9}, the maximum traffic load (ρ) per input must be 2.67×10^{-9} or less with four inputs and four outputs. This is equivalent to having fewer than four packets contain data out of ten million packets arriving at an input. For an ATM cell stream at OC-192 rates (9.95 Gbps) fewer than four cells can be sent in per minute. This is obviously a very inefficient use of the available bandwidth in the switch architecture. Table 3.1 shows the maximum allowed traffic load per input for various PLP levels and numbers of inputs and outputs. It is interesting to note that the PLP performance of the crossbar switch does not vary significantly with the number of inputs and outputs for square switches when the traffic load is light, unlike the visible difference in the results for heavier loads in figure 3.4 above.

The PLP performance does get slightly worse as the number of inputs increases, as expected because of the higher probability of packet collisions.

With two inputs, the probability of two packets colliding at a single output is the probability of a single packet arriving at each input destined for the same output (equation 3.2, see also Appendix B). The probability of a packet arriving at an input for a specific output is ρ/2; one packet is dropped when there is a collision, and the number of different ways two packets can arrive is 1.

$$P_{collide\ 2x2} = C_2^2 \cdot \left(\frac{\rho}{2}\right)^2 \cdot \left(1-\frac{\rho}{2}\right)^0 = \frac{\rho^2}{4} \qquad \text{Eq. 3.2}$$

With three inputs, the probability of two or more packets colliding at a single output is the probability of two packets arriving at two of the three inputs destined for the same output plus the probability of three packets arriving at the inputs all destined for the same output (equation 3.3, see also Appendix B). The probability of a packet arriving at an input for a specific output is ρ/3, the number of dropped packets is 1 or 2 depending on how many packets collide, and the number of different combinations of inputs that two packets can arrive is 3 (3 choose 2). The number of different combinations of inputs that three packets can arrive at is 1.

$$P_{collide\ 3x3} = C_2^3 \cdot \left(\frac{\rho}{3}\right)^2 \cdot \left(1-\frac{\rho}{3}\right)^1 + 2 \cdot C_3^3 \cdot \left(\frac{\rho}{3}\right)^3 \cdot \left(1-\frac{\rho}{3}\right)^0 = \frac{\rho^2}{3} - \frac{\rho^3}{27} \qquad \text{Eq. 3.3}$$

Also as expected, when the number of outputs is larger than the number of inputs the allowable traffic load increases because of the lower chance of packets colliding at an output, while the reverse is true when there are more inputs than outputs.

	PLP of 10^{-8}	PLP of 10^{-9}	PLP of 10^{-10}
N=M=4	2.67×10^{-8}	2.67×10^{-9}	2.67×10^{-10}
N=M=8	2.29×10^{-8}	2.29×10^{-9}	2.29×10^{-10}
N=M=16	2.13×10^{-8}	2.13×10^{-9}	2.13×10^{-10}
N=4, M=8	4.00×10^{-8}	4.00×10^{-9}	4.00×10^{-10}
N=8, M=4	1.14×10^{-8}	1.14×10^{-9}	1.14×10^{-10}

Table 3.1 - Crossbar Switch Maximum ρ for Desired PLP

Knockout Switch

The knockout switch uses a form of output buffering. In its original form [Yeh87] the knockout switch has L multiple paths from any input to any output, requiring N x (M x L) crosspoints. A block diagram of the knockout switch is shown in figure 3.5 with L set to three. This switch architecture can then deliver up to L packets to any output in a single time slot, which requires a buffer at each output. The switch architecture is based on L being less than N (the pure output buffering case), but still large enough to meet the desired packet loss performance. The number of paths to an output, L, and the output queue size, Q_O, are determined from the characteristics of the arriving traffic and the desired PLP and latency performance. If more than L packets are destined for the same output in a single time slot, the extra packets are lost or 'knocked out' of the switch. Packets are also lost if the output buffers overflow. Packets are kept in sequence automatically through this switch because of the output buffering. Since a packet may be delayed in the output queue for some time the latency of this switch, although optimal, is not zero. The latency is optimal because the output buffering only delays a packet for other packets destined for the same output.

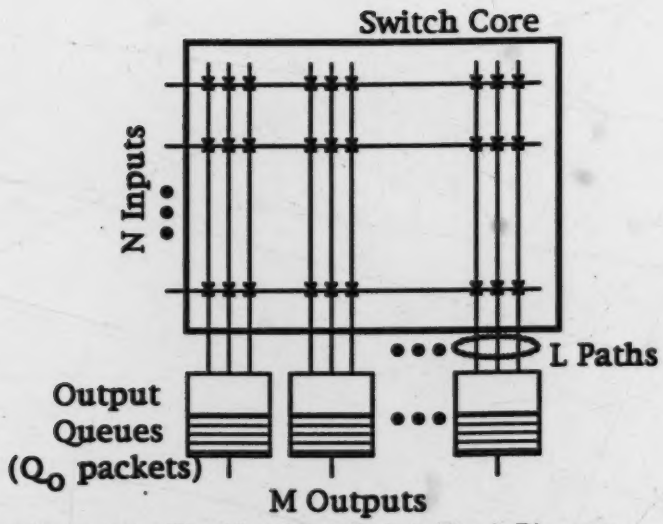


Figure 3.5 - The Knockout Switch Block Diagram

In its original form the knockout switch is based on electrical constraints, under which it is easier to add additional paths to an output than to increase the bit rate of a single path. A logical equivalent of the knockout switch, optimized to reduce the number of crosspoints, has the core of the switch operating L times faster than the inputs and outputs and only requiring N x M crosspoints. Each crosspoint must be capable of transmitting L times the input bit rate. The output queue still must be capable of accepting L packets in a single time slot. As well, an input queue is required to buffer the arriving packet and transmit it into the switch core at the higher bit rate. There is an optimum value, in terms of cost, for the number of paths to an output and the switch core speedup that varies between 1 and L that depends on the technology used to implement the knockout switch. The control algorithm for the knockout switch is almost as simple as that for the crossbar switch, and is given in figure 3.6.

```
                Knockout Switch Control Algorithm

for each time slot
    for each input
        if a packet is not arriving at this input then
            goto next input
        endif
        for each of L paths to desired output
            if this path to desired output is free then
                route the packet along this path
                goto next input
            endif
        next path to the desired output
        the packet is blocked so drop it
    next input
next time slot
```

Figure 3.6 - Knockout Switch Control Algorithm

A theoretical analysis of the knockout switch under uniform loads is given in Appendix C, and results are shown in figure 3.7. Simulations were not performed on this switch architecture. This switch architecture only needs FIFO buffers for the output queues to keep the packets in sequence, as described earlier in the section on simple output buffering. For ATM cells a small 512x18 bit FIFO buffer, such as the PDM42215 from Paradigm [Qui94] can be considered. It holds 21 cells with an access time of 10 nsec. Using one of these FIFO buffers per output or sharing one among two outputs results in output queue sizes of 21 and 10 packets respectively.

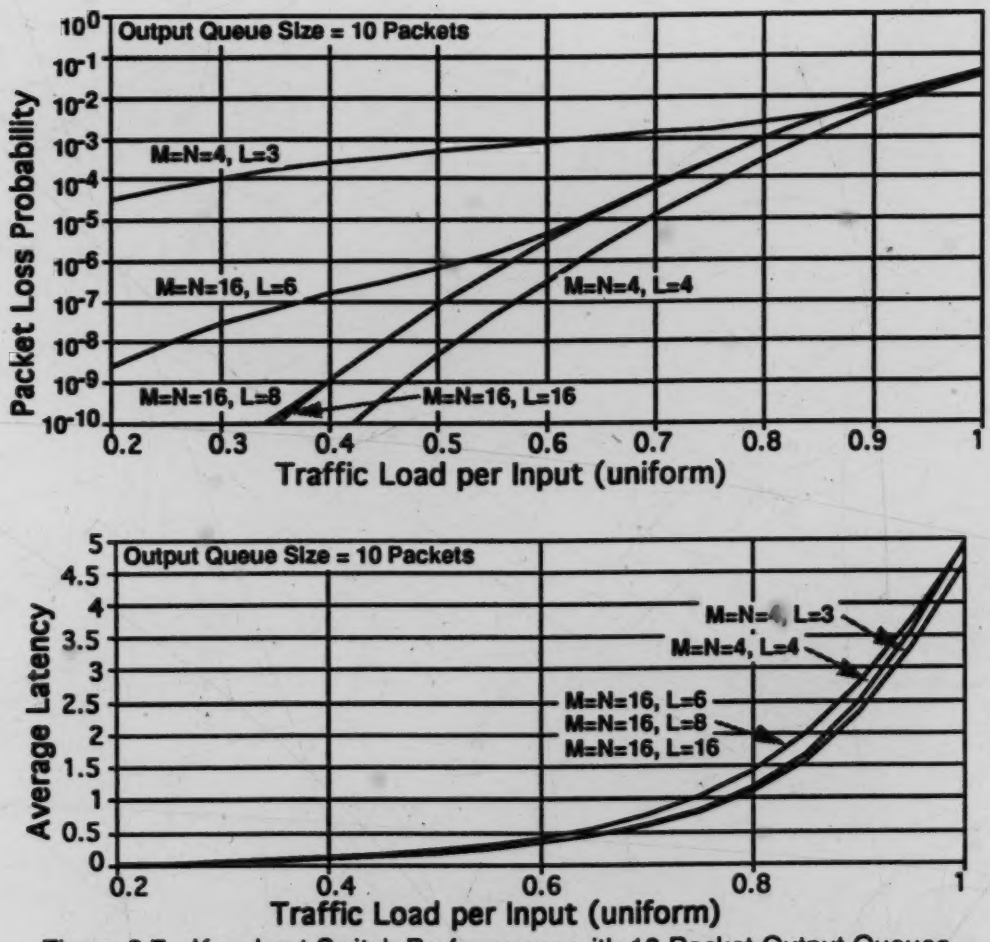


Figure 3.7 - Knockout Switch Performance with 10 Packet Output Queues

When the number of paths to an output, L, equals the number of inputs, N, the knockout switch is equivalent to a simple output buffering switch and packets are only lost when the output queue overflows. At the other extreme, when L is equal to one, the knockout switch reduces to a crossbar switch. As seen in both figures 3.7 and 3.8, for a 4x4 square switch L must be equal to the number of inputs to have a reasonably low PLP. For a larger 16x16 switch, L can be as low as 9 while still achieving a PLP of 10^{-9} at a traffic load of 0.6 per input with a queue size of 21 packets. This is a significant reduction from the simple output buffering case where L is set to 16. The performance of the

knockout switch is much better in larger switches where the extra paths (beyond L) to an output in simple output buffering are seldom used, so this switch architecture scales easily to larger switches. The average latency of the knockout switch is largely controlled by the output queue size and it changes very little when the number of inputs and outputs are changed.

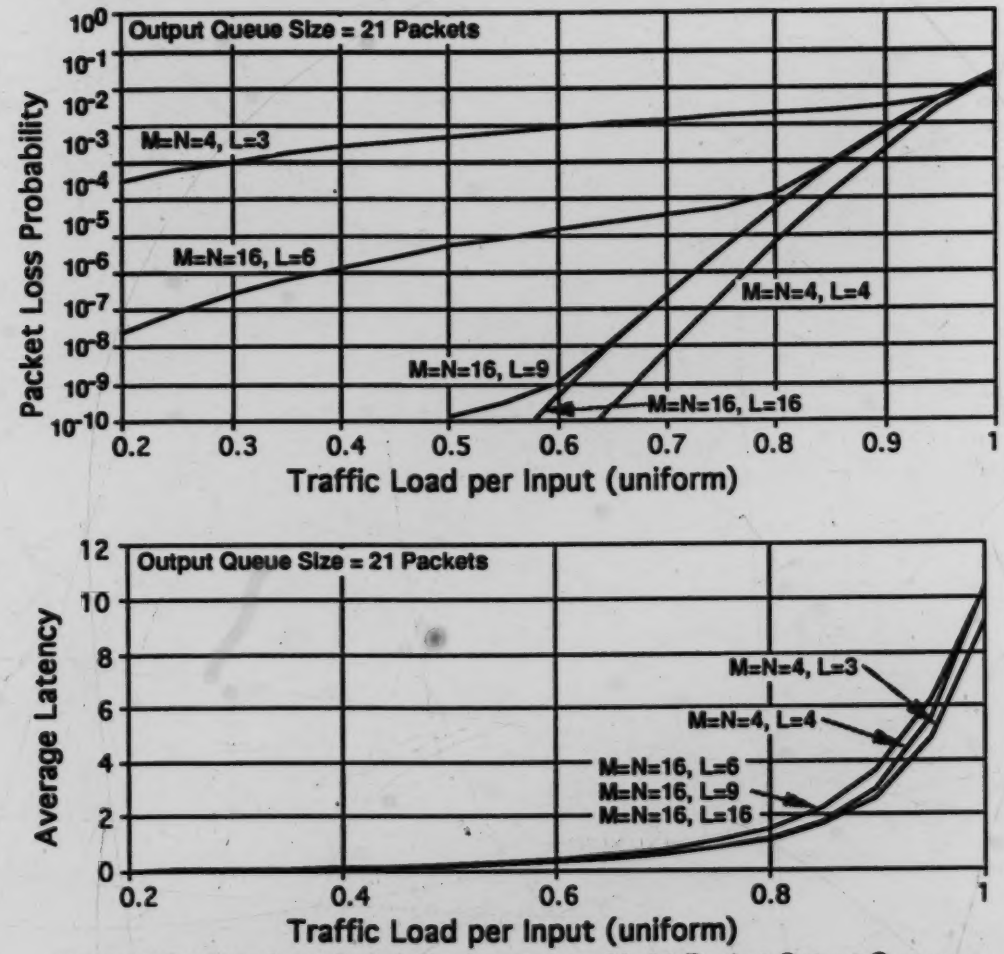


Figure 3.8 - Knockout Switch Performance with 21 Packet Output Queues

From figures 3.7 and 3.8 it can be seen that there are two distinct mechanisms for packet loss in the knockout switch. At low traffic loads packet collisions, where more than L packets arrive in the same time slot destined for the same output, dominate the PLP performance of the knockout switch. At high

traffic loads output queue overflow dominates the packet loss performance of the switch. As table 3.2 shows, increasing the output queue size from 10 to 21 packets reduces the PLP of the knockout switch by 3 orders of magnitude if the switch is operating where output queue overflow is the dominant packet loss mechanism.

	Q_O = 10 Packets	Q_O = 21 Packets	Dominant PLP Mechanism
N=M=4, L=3, ρ=0.6	1×10^{-3}	1×10^{-3}	Packet Collision
N=M=4, L=4, ρ=0.7	1×10^{-5}	1×10^{-8}	Queue Overflow
N=M=16, L=6, ρ=0.4	2×10^{-7}	2×10^{-6}	Approximately Equal
N=M=16, L=9, ρ=0.6	2×10^{-6}	$< 10^{-9}$	Queue Overflow

Table 3.2 - Knockout Switch PLP Change with Output Queue Size

Staggering Switch

The staggering switch [Haa92b] essentially splits the switch into two crossbar matrices (of NxP and PxM crosspoints each) connected with P fiber delay line buffers. These P fiber delay lines act as FIFO buffers, with each storing a different number of packets ranging from 1 to P. The first section ('scheduling' stage) of the switch schedules the incoming packets from the N inputs over the intermediate P buffers of varying delays so that no two packets destined for the same output arrive in the same time slot at the second section of the switch. This requires that only one packet is destined for a particular output in any one column as shown in figure 3.9. The second section ('switching' stage) of the switch then simply sends the incoming packets arriving from the P buffers to their desired output out of the M possible outputs.

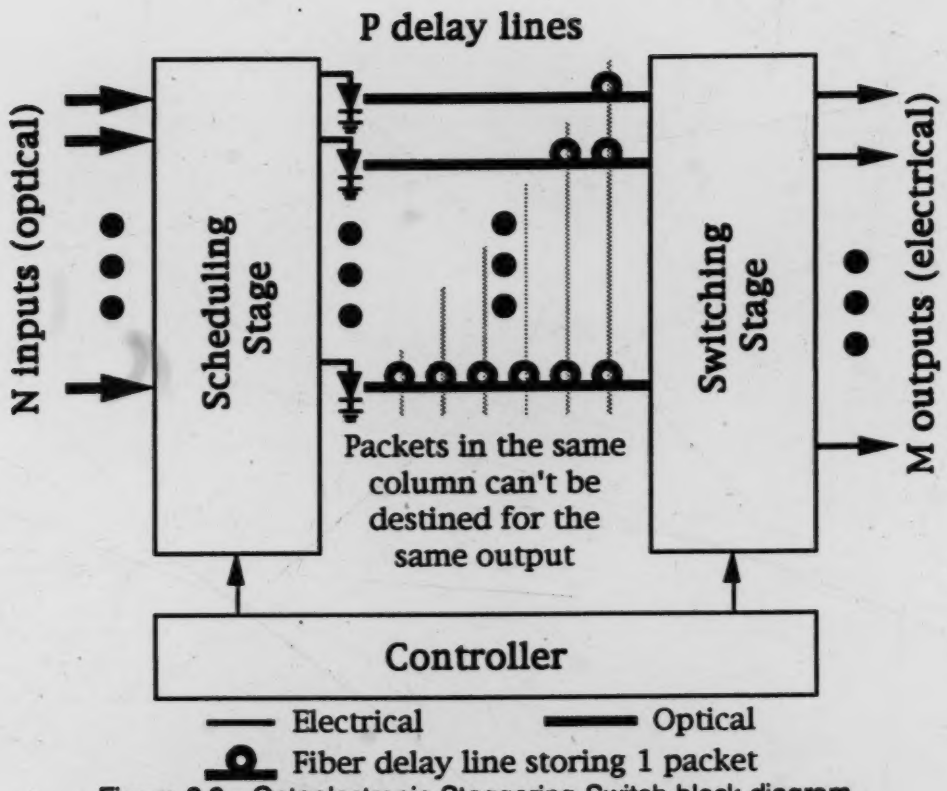


Figure 3.9 - Optoelectronic Staggering Switch block diagram

The staggering switch adds buffering to decrease the packet loss probability, unfortunately this added buffering also increases the latency of the switch. The buffers between the stages are shared among all of the outputs, which is the optimal buffering method, except that the fiber delay lines act as a FIFO memory rather than a RAM. This means that even if an output is free, and a packet destined for that output is in one of the buffers it will not reach the switching stage in this time slot, and will not be delivered. This adds extra delay above the optimal delay delivered by shared output queuing with a RAM. The upper bound on the latency of the switch is obviously the delay of the longest interconnecting buffer. The longest buffer must hold at least as many packets as the queue size required when using simple output buffering to achieve the same PLP performance. There is no buffer that feeds back to the inputs so

there is no possibility of looping indefinitely and each packet undergoes exactly two O/E and two E/O conversions in passing through the switch (assuming electrical in, electrical out) leading to a constant signal to noise ratio (SNR) for the packet. This simplifies the design of the optical receiver used in each stage. Since a limited number of conversions are required, it may be possible simply to amplify the signal to drive the laser, rather than requiring regeneration of the digital signal. This switch has a blocking architecture (it is blocked if interconnecting buffers are all full) and also has a chance of packet collision (at the scheduling stage rather than the output): the packet loss probability is the sum of these two probabilities. The staggering switch does not preserve packet order under a simple first-come first-served control algorithm such as that shown in figure 3.10. The probability of packets arriving at their destination out of sequence when this type of controller is used is quite high, shown in figure 3.11, at about 1 packet in 1000 out of sequence even with a very light traffic load.

```
            Simple Staggering Switch Control Algorithm

                          Scheduling Stage
for each time slot
      for each input
            if a packet is not arriving at this input then
                  goto next input
            endif
            for each buffer
                  if buffer first slot free then
                        if desired output is not in conflict with another packet
                              route the packet to this buffer
                              goto next input
                        endif
                  endif
            next buffer
            the packet is blocked so drop it
      next input
next time slot

                          Switching Stage
for each time slot
      for each buffer
            if a packet is not arriving from this buffer then
                  goto next buffer
            endif
            route the arriving packet to its desired output
      next buffer
next time slot
```

Figure 3.10 - Simple Staggering Switch Control Algorithm

One of the problems with this switch is that the 'transmission time' of a packet is set when the packet is placed in a buffer and then cannot be changed. This leads to an inability to implement packet priority schemes, where a high priority packet can preempt a low priority packet that arrived earlier. In the staggering switch architecture, once a packet is routed into a fiber delay line, it will arrive at the output a set time later. If a higher priority packet wants to preempt a packet that is already in the delay line, the packet in the delay line will be lost. A better switch architecture would allow the low priority packet to be delayed rather than lost. The other problem is that a considerable amount of the available buffer space is wasted in the operation of the staggering switch. Since P must be greater than or equal to N for acceptable packet loss

probability and at most N packets will be routed into the P buffers in a single time slot, P-N buffer spaces will be wasted in each time slot. As well, the P buffers have a total of $0.5x(P+P^2)$ buffer spaces available, but the arriving packets can only be assigned to one of P buffer spaces that are at the beginning of the delay lines. This means any empty buffer spaces that exist in the middle of the fiber delay lines cannot be accessed and are wasted.

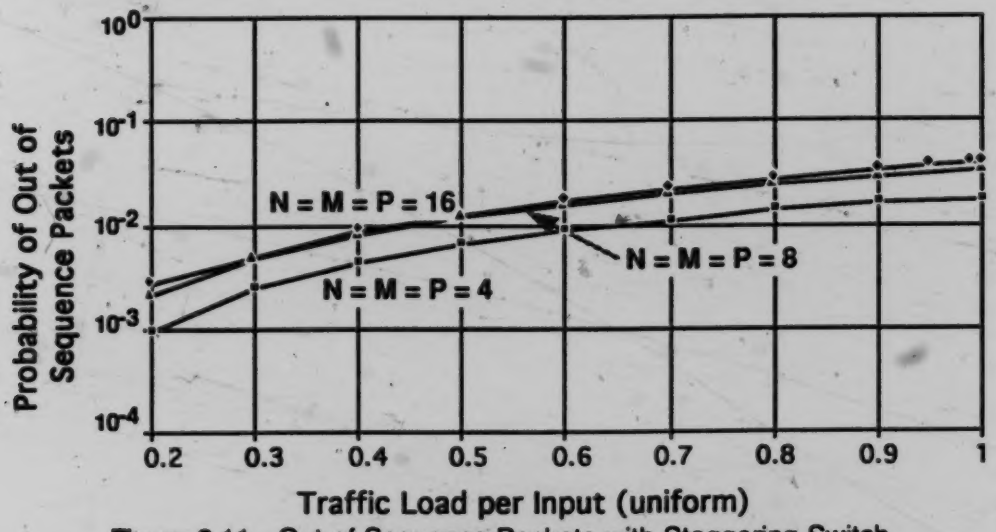


Figure 3.11 - Out of Sequence Packets with Staggering Switch

If all the buffers in the staggering switch have the same delay, then the staggering switch does preserve packet order but this configuration is the equivalent of a simple crossbar switch with extra delay and O/E and E/O conversions added. The sensible method to preserve packet sequence is to use a more complex controller which tracks the packets that are in the buffers and schedules each arriving packet so it doesn't arrive at the second stage out of sequence.

There are three packet tracking methods possible: the first is to track by the input the packet arrived on; the next is to track by the packet's destination; and, finally, to track the packet based on the input it arrived at and its desired

destination. The first method requires the controller to track the buffer position of the latest packet to arrive from a particular input. This adds to the complexity of the controller by forcing it to track N pointers, and route the arriving packet from an input into a buffer that has a longer delay than the remaining delay of the latest packet from the same input. The second method is similar to the first, except for using M pointers to track the buffer position of the latest packets based on their desired output. The third method requires the controller to use NxM pointers that track the buffer position of the latest packet that arrived for the specific input/output combination as shown in figure 3.13. This requires a much more complex controller, as the algorithm in figure 3.12 shows. Since each of these methods preferentially assigns arriving packets to buffers with ever longer delays, the packet loss probability and average latency of the switch will increase. As can be seen in figure 3.13, when packets are sequenced based on input or output alone, the packet loss probability increases considerably. However, when the packets are sequenced based on the input and output combination, there is only a very slight increase in the packet loss probability.

```
Sequence Preserving Staggering Switch Control Algorithm

                    Scheduling Stage
for each time slot
    for each input
        if a packet is not arriving at this input then
            goto next input
        endif
        for each buffer
            if this buffer's length is less than position[input][output] then
                goto next buffer
            endif
            if buffer first slot free then
                if desired output is not in conflict with another packet
                    route the packet to this buffer
                    position[input][output] set to this buffer's length
                    goto next input
                endif
            endif
        next buffer
        the packet is blocked so drop it
    next input
    for every input and output combination
        if position[input][output] > 0 then
            decrement position[input][output] by one
        endif
    next input and output combination
next time slot

                    Switching Stage
for each time slot
    for each buffer
        if a packet is not arriving from this buffer then
            goto next buffer
        endif
        route the arriving packet to its desired output
    next buffer
next time slot
```

Figure 3.12 - Sequence Preserving Staggering Switch Control Algorithm

When examining sequence preserving staggering switches of various sizes as shown in figure 3.13, it is obvious that the PLP performance of a larger switch is much better than that of a smaller switch with a proportionate number of buffers. This indicates that this switch architecture will scale to larger sizes easily, ignoring technology limitations. However, the average (and maximum) latency of the larger switch worsens.

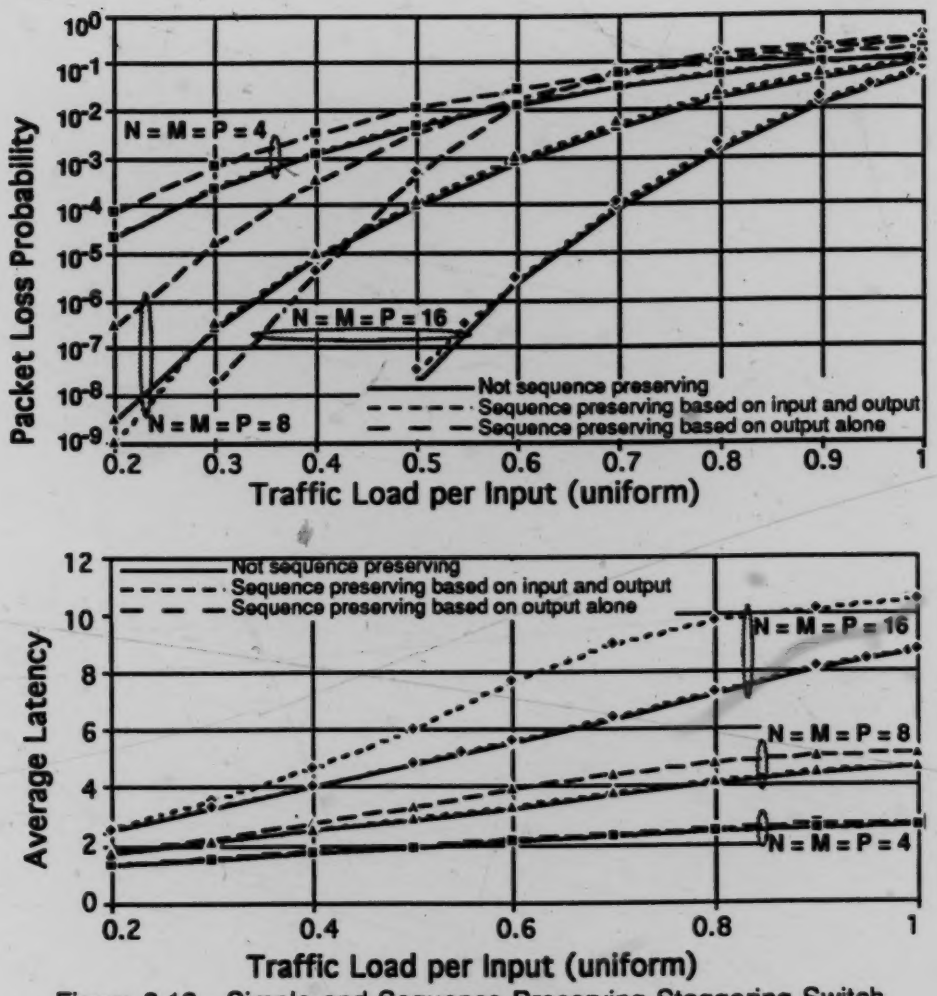


Figure 3.13 - Simple and Sequence Preserving Staggering Switch Performance

The staggering switch can achieve an arbitrarily low packet loss probability for a given traffic load simply by increasing the number of fiber delay lines connecting the two stages of the switch as shown in figure 3.14. Again, larger switches improve their PLP performance more than smaller switches for the same relative change in the number of buffers. However, adding more delay lines increases the number of crosspoints required in each stage, as well as increasing the average and maximum latency of the switch.

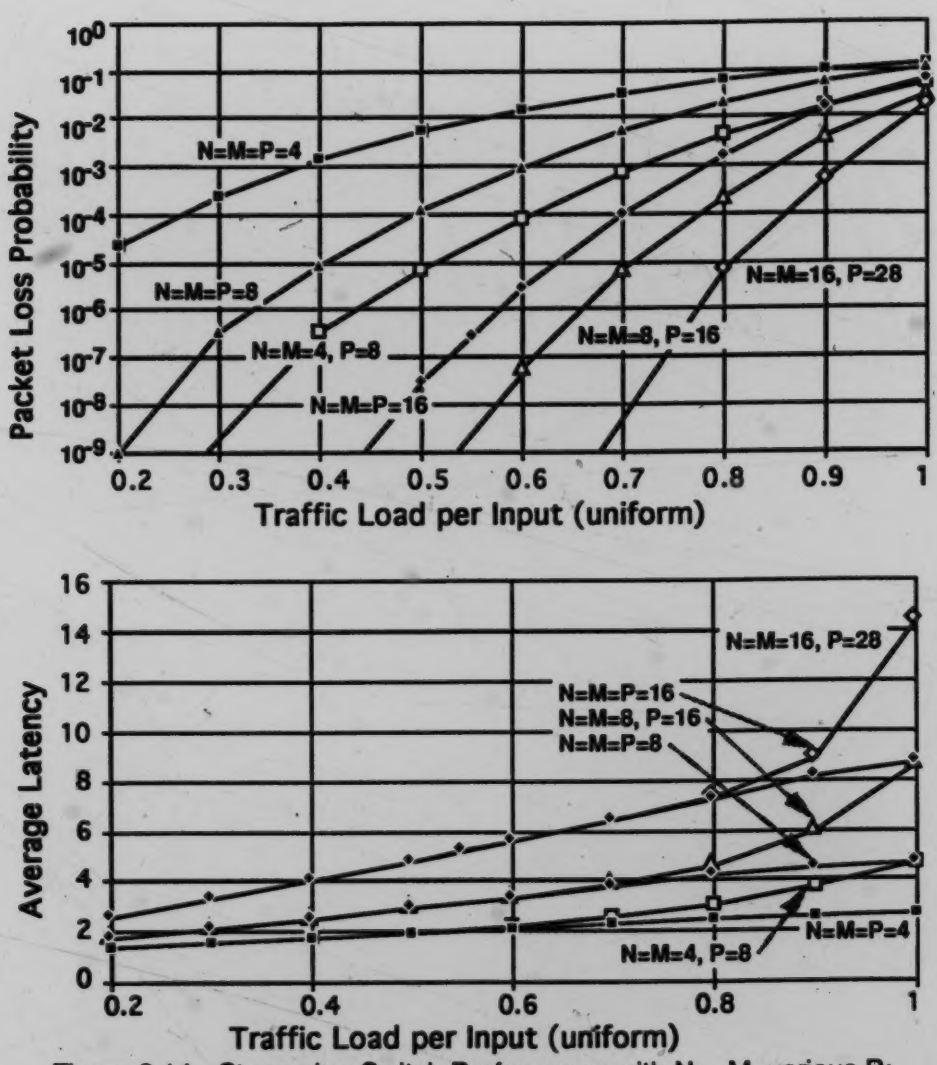


Figure 3.14 - Staggering Switch Performance with N = M, various P;

Reflex Switch - Multiple Packet Buffers

The reflex switch loops fiber delay line buffers from P extra outlets to P extra inlets (for a total of (N+P)(P+M) crosspoints) and allows the packets to recirculate indefinitely in these buffers as the block diagram in figure 3.15 shows. This architecture, using lengths of fiber as delay lines and GaAs photoconductors as crosspoints, was first proposed by D. Lam and R.I. MacDonald [Lam84] as a reconfigurable delay line. The use of the reflex

architecture as a fast packet switch was developed independently by both the author and M.J. Karol [Kar93] in approximately the same time frame. Karol's version of the reflex switch was optimized for an optical switch, in which each crosspoint is very expensive and not easily integrated. Because of this, Karol's version of the switch is optimized to achieve the best PLP performance for a given number of crosspoints. This results in Karol's switch using the same buffer structure as the staggering switch (termed staggered buffers). The reflex switch uses photodetectors as the crosspoints, resulting in a much simpler and less expensive crosspoint that is relatively easy to integrate. This single fact changes the constraints placed on the switch architecture significantly, as this section and the next will demonstrate.

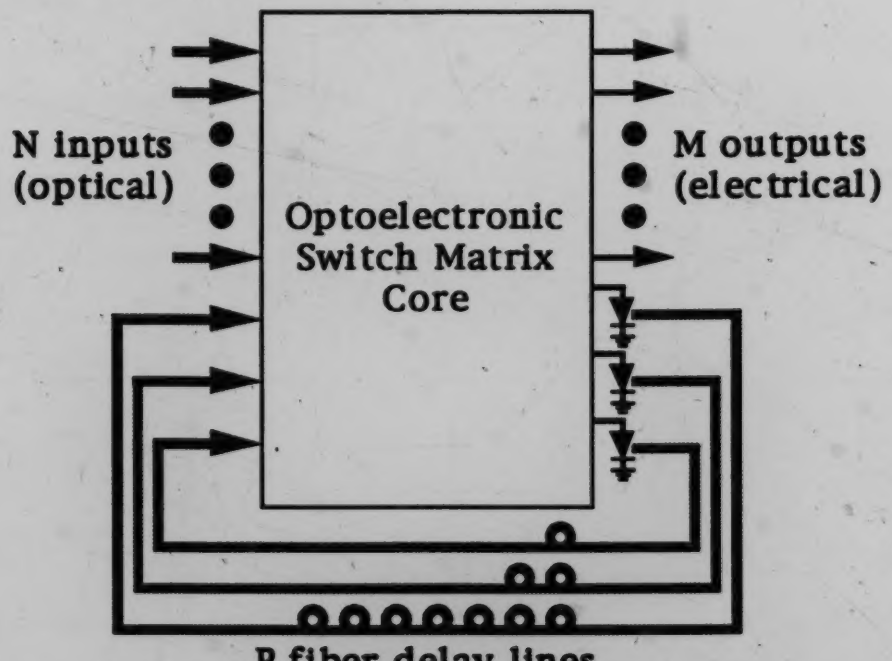


Figure 3.15 - Optoelectronic Reflex Switch Block Diagram

The reflex switch is a superset of the staggering switch in that the reflex switch is formed by connecting the two stages of the staggering switch and adding extra crosspoints to allow the inputs to connect directly to the outputs

and the buffer outputs to connect directly to the buffer inputs. The electronic controller, not shown in figure 3.15, will send an incoming packet either to its desired output or to a buffer if the desired output is already busy. Because it uses FIFO buffers in the same way as the staggering switch, for a given buffer structure, the reflex switch will always perform at least as well as the staggering switch. Packets are allowed to proceed directly (i.e. without having to go through a delay line) to their desired output if it is free which reduces the average latency below that of the staggering switch under normal conditions. The staggering switch can also be made to have zero low-load latency by providing N zero-delay connections between the switching and scheduling stages. This increases the number of crosspoints so that the reflex and staggering switch become equivalent for low loads.

Under high traffic loads, packets can circulate through the buffers more than once, decreasing the PLP but increasing the average latency. Throughout this section, the reflex switch is assumed to use staggered buffers, that is P fiber delay lines capable of storing a different number of packets from 1 to P respectively. The total buffer space available in this architecture with this buffer structure is $0.5x(P+P^2)$ packets.

```
Non-Sequence Preserving Reflex Switch Control Algorithm

for each time slot
    for each input (from buffers first, then from outside world)
        if its desired output is free then
            route the packet to its desired output
        else
            for each buffer
                if buffer first slot free then
                    route the packet to this buffer
                    goto next input
                endif
            next buffer
            drop the packet (collision)
        endif
    next input
next time slot
```

Figure 3.16 - Non-Sequence Preserving Staggered Buffer Reflex Switch Control Algorithm

Again, in this switch architecture buffers have been added around the simple crossbar switch to decrease the packet loss probability and these buffers increase the latency of the switch. The reflex switch is a strictly non-blocking architecture (a free input and output can always be connected). A packet is lost only when two or more packets are destined for the same output and there are no free buffers available. In general, the reflex switch does not preserve packet sequence although the simple matter of serving the buffer inlets before the inputs helps to decrease both the maximum latency and the number of out of sequence packets. Figure 3.18 shows the simulated probability of packets arriving at their destination out of sequence. Compared to the staggering switch, the reflex switch has more out of sequence packets at high traffic loads due to the recirculating buffers, and fewer out of sequence packets at low traffic loads because of the ability to send a packet directly to its desired output. In the same manner as the staggering switch, a more complex controller than the simple first come-first served algorithm can be used to preserve packet sequence at the expense of packet loss probability and latency. Both controller

algorithms are shown in figures 3.16 and 3.17, with a performance comparison presented in figure 3.19. Since a packet may travel through a variable number of buffers between 0 and P, the signal to noise ratio (SNR) at the receiver is not constant which requires careful receiver design to keep the bit error rate (BER) of the signal low. This may be alleviated by regenerating the bit stream at each buffer, which would also simplify the problem of keeping all the packets synchronized.

Sequence Preserving Reflex Switch Control Algorithm

```
for each time slot
    for each input
        if a packet is not arriving at this input then
            goto next input
        endif
        for each buffer
            if this buffer's length is less than position[input][output] then
                goto next buffer
            endif
            if buffer first slot free then
                if desired output is not in conflict with another packet
                    route the packet to this buffer
                    position[input][output] set to this buffer's length
                    goto next input
                endif
            endif
        next buffer
        the packet is blocked so drop it
    next input
    for every input and output combination
        if position[input][output] > 0 then
            decrement position[input][output] by one
        endif
    next input and output combination
next time slot
```

Figure 3.17 - Sequence Preserving Staggered Buffer Reflex Switch Control Algorithm

The reflex switch, when used with staggered buffers, does implement fully shared output buffering. Unfortunately, as in the staggering switch, the buffers are not randomly accessible. However, the reflex switch does allow packets to travel from one buffer to another, which allows packet priority schemes to be implemented without causing the low priority packet be dropped,

which is an improvement over the staggering switch. The 'transmission time' of packets is updated when a packet enters or exits a buffer, although still not at every time slot (as it would in a true random access scheme). If a low priority packet is to be preempted, it can be sent into another buffer, delaying it for one or more time slots, while the high priority packet is sent to the desired output instead. Buffer spaces are still wasted as a packet may be in the middle of a delay line when its desired output is free, delaying the packet unnecessarily and decreasing the throughput of the switch.

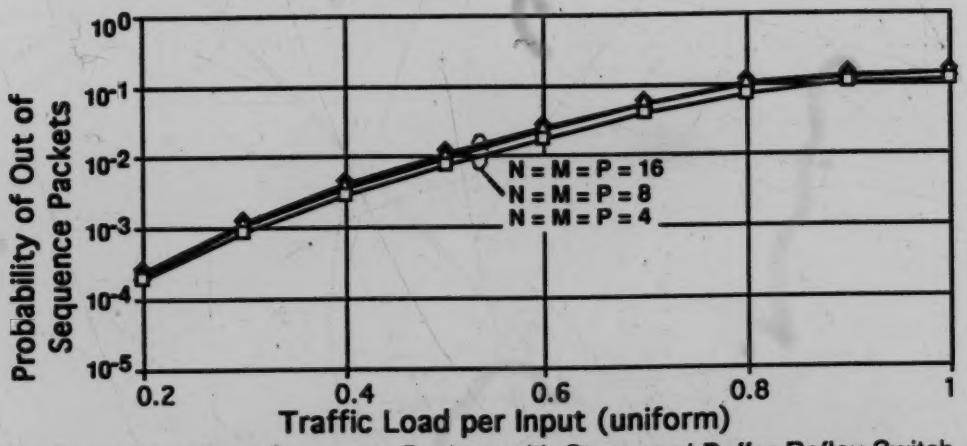


Figure 3.18 - Out of Sequence Packets with Staggered Buffer Reflex Switch

Initially, it would seem that a packet can recirculate indefinitely within the reflex switch. However, if the controller algorithm serves the buffers in a consistent order before the inputs, the buffers are treated preferentially in much the same way that the first input is served preferentially in the absence of the shuffling stage of the controller algorithm. This means that the first buffer will always get its desired output, the second buffer will always get its desired output or be connected to the first buffer (if the second buffer's desired output was the same as the first), etc. At the next time slot, this process is repeated, so that eventually a packet that is in any buffer will end up in the first buffer and be sure of being sent to its desired output in the next time slot. Of course, the packet

might be switched to its output much earlier when the output happens to be free. This process gives a strict upper bound on the maximum latency a packet can experience within the reflex switch equal to the sum of all the delays in all the buffers which is $0.5x(P+P^2)$. A packet will only experience the maximum latency when all the other buffers are completely filled with other packets destined for the same output, unlike the staggering switch where the maximum latency is experienced whenever a packet is sent through the longest fiber delay line, which will happen much more often.

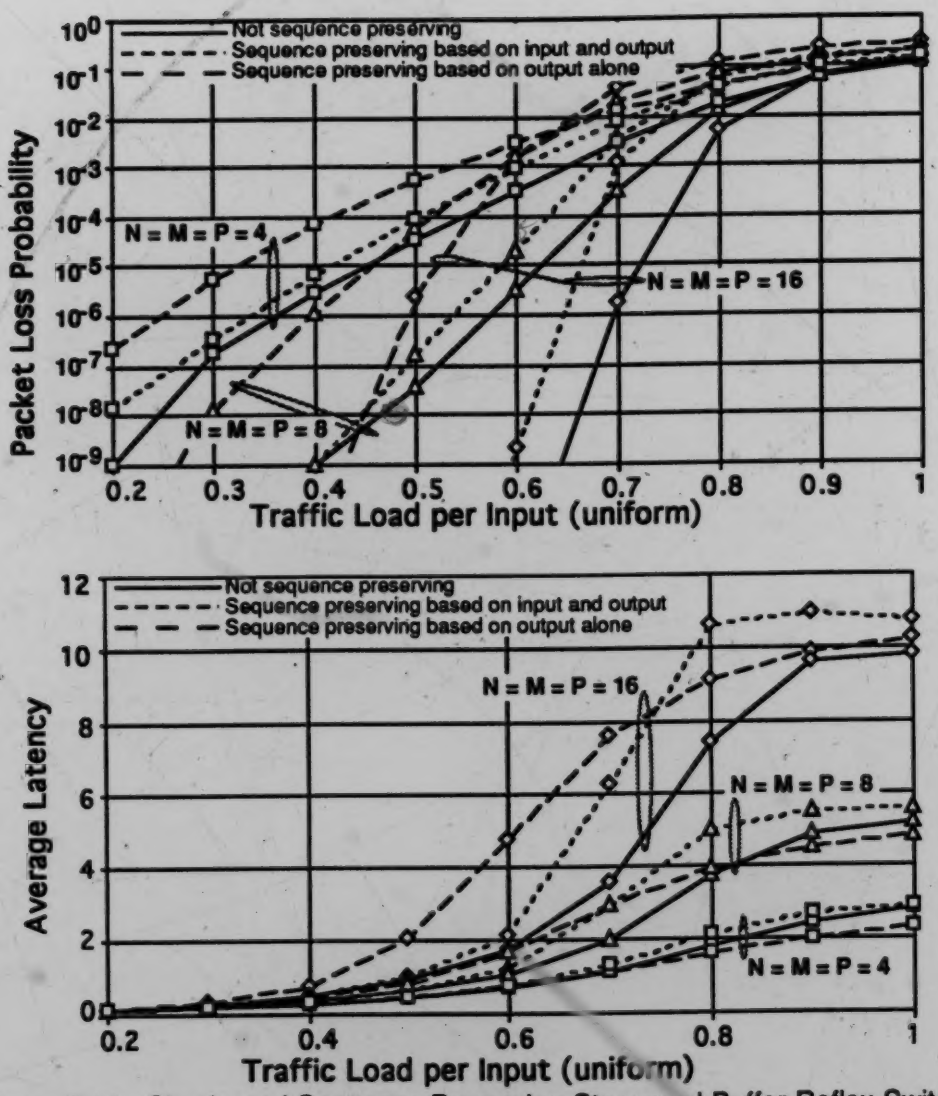


Figure 3.19 - Simple and Sequence Preserving Staggered Buffer Reflex Switch Performance

Assuming similar buffer configurations, the reflex switch will achieve lower packet loss probability and latency than the staggering switch, at the expense of controller complexity due to the increased number of crosspoints. This can be intuitively seen. Those packets in the reflex switch that are switched directly to their outputs (not allowed in the staggering switch) will decrease the average latency of the reflex switch. These packets will not

require buffer space, since they have been delivered to their desired output, leaving more buffer space for other packets, decreasing the packet loss probability of the reflex switch. The results shown in figure 3.19 demonstrate this, as well as the scalability of the staggered buffer reflex switch to larger sizes. It is interesting to notice the large increase in average latency at higher traffic loads where packets recirculate through the fiber delay lines more than once.

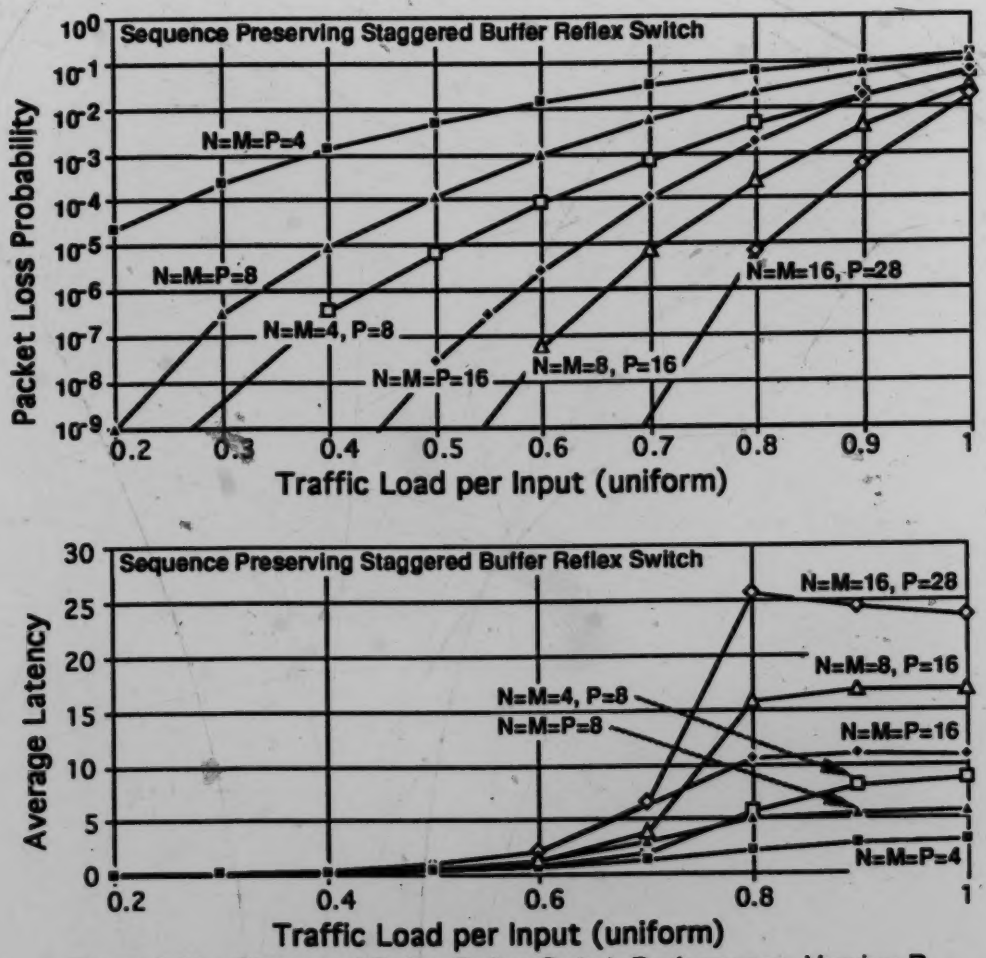


Figure 3.20 - Staggered Buffer Reflex Switch Performance, Varying P

As figure 3.20 shows, increasing the number of buffers, P, decreases the PLP of the switch in the same manner as the staggering switch. Any desired

PLP performance level can be achieved by selecting the appropriate number of fiber delay lines. Again, for the same relative change in the number of buffers, a larger switch demonstrates more improvement in PLP performance than a smaller switch. Unfortunately, increasing the number of fiber delay lines increases the average and maximum latency of the switch, increases the number of crosspoints, and makes the electronic controller more complex.

Reflex Switch - Single Packet Buffers

By restricting the fiber delay lines in the reflex switch to only store one packet each, as shown in figure 3.21, the reflex switch is formed into a multi-ported serial access RAM capable of storing P packets. Each input to the switch acts as a write port, each output becomes a read port, and each buffer becomes a memory location. This serial access RAM is ideally suited for use a switch because of its ability to store a packet in a fiber delay line and at the same time read a different packet out of the same fiber delay line without a conflict since the accesses are serial. The N input ports to the RAM allow up to N incoming packets to be stored during a single time slot without requiring the access time to be $1/N^{th}$ of a time slot. The M output ports also allow up to M packets to be read out of the RAM without requiring an extremely fast access time. However, the amount of available buffering decreases to P packets from the $0.5x(P+P^2)$ packets available with the staggering-switch style buffers. To have the same buffering capacity as the staggering switch requires many more fiber delay lines, which increases the number of crosspoints required to implement the switch. Each of the buffering spaces is fully utilized however, with none of the buffer spaces wasted as happens with the staggered buffer structure. The reflex switch architecture using single packet buffers will be referred to as the simple reflex switch.

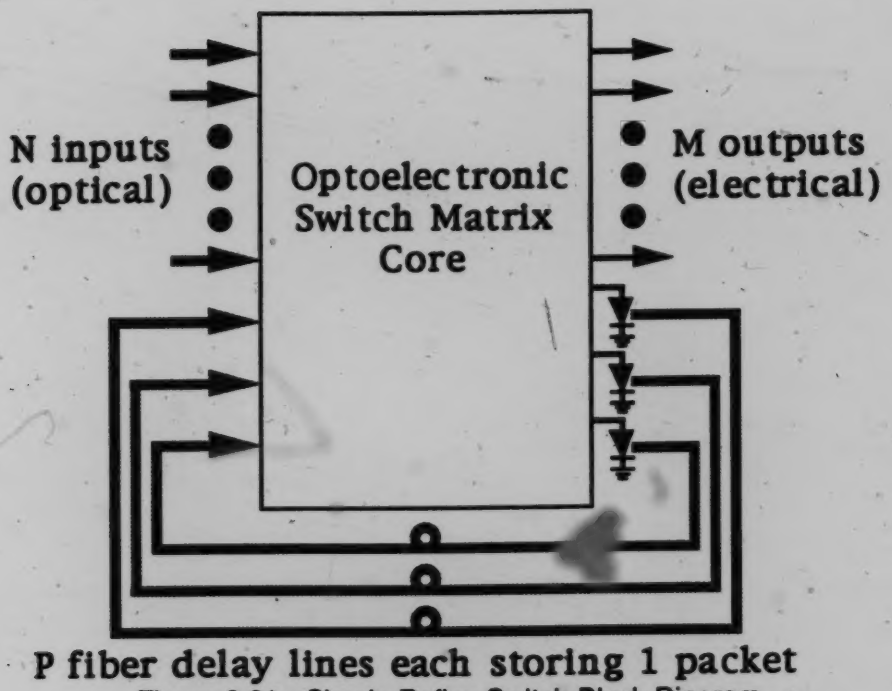


Figure 3.21 - Simple Reflex Switch Block Diagram

The tremendous advantage this buffering scheme has over the staggering switch buffer structure is that packet sequence is automatically preserved, even when using a simple first come - first served controller algorithm such as that shown in figure 3.22. As long as the buffers are served in a consistent order before the inputs, the packet sequence will be preserved. By serving the buffers in a consistent preferential manner the buffers are assigned priorities, with the first buffer served having the highest priority because the packet in the first served buffer will always be delivered to its desired output. Similarly, the packet in the second buffer served will always be delivered either to its desired output, or to the highest priority buffer. If the packet was sent to the highest priority buffer, it is guaranteed to be delivered to its output in the next time slot. In this manner each arriving packet is only delayed by those packets that are destined for the same output. Since the controller must set the state of each of the crosspoints in every time slot, a simple control algorithm allows the

bit rate of the inlets and outlets to be increased, or the number of inlets and outlets to be increased, or both.

```
   Simple Sequence Preserving Reflex Switch Control Algorithm

for each time slot
     for each input (from buffers first, then from outside world)
          if its desired output is free then
               route the packet to its desired output
          else
               for each buffer
                    if buffer first slot free then
                         route the packet to this buffer
                         goto next input
                    endif
               next buffer
               drop the packet (collision)
          endif
     next input
next time slot
```

Figure 3.22 - Simple Sequence Preserving Reflex Switch Control Algorithm

As pointed out by Karol [Kar93], the simple reflex switch architecture is not suitable for use with optical crosspoints, as extra crosspoints are required to achieve the same PLP performance as achieved with the staggered buffers. However, photodetector crosspoints are much less expensive and their use justifies the consideration of this architecture. Photodetector crosspoints, as well as optical crosspoints, can have incredibly high speeds (full width half maximum impulse response times as low as 0.87 psec [Cho92] when manufactured with low temperature grown GaAs). Simple GaAs metal - semiconductor - metal photodetectors (MSM PD) that are very easy to integrate with the electronic receiver are available with 3 dB bandwidths of 11 GHz [Hur91]. Bandwidths as high as these easily allow OC-192 bit rates of approximately 10 Gbps. An electronic controller, used in both the optical and optoelectronic versions of the reflex switch, will only have 42 nsec to determine the crosspoint states for the entire switch at this bit rate. Obviously, the electronic controller will be the bottleneck in this switch architecture, whether

using optical or optoelectronic crosspoints, making a simpler algorithm which can operate more quickly a distinct advantage.

With this buffering scheme, every packet gets a chance at its desired output in every time slot. This means that there are absolutely no wasted buffer spaces and the latency of the switch is optimal. All the P buffers can be used by all the outputs, so this switch architecture implements fully shared output buffering. All buffered packets can have their 'transmission time' updated in every time slot, since every buffered packet leaves a buffer in every time slot, allowing for the simple implementation of packet priority schemes.

The disadvantage of this buffer structure is that there are only P buffer spaces available when P buffers are used. This is considerably less than the $0.5x(P+P^2)$ buffer spaces available when using the staggered buffers. Because of this, the simple reflex switch requires more buffers, and therefore more crosspoints, to achieve the same PLP performance as either the staggering switch or the reflex switch with staggered buffers. Unlike the staggered buffer case, with single packet delay lines all of the P available buffer spaces are fully utilized with none being wasted. This means that the simple reflex switch can achieve the same PLP performance with fewer buffer spaces, although more buffers, than required by the switch architectures using the staggered buffer structure.

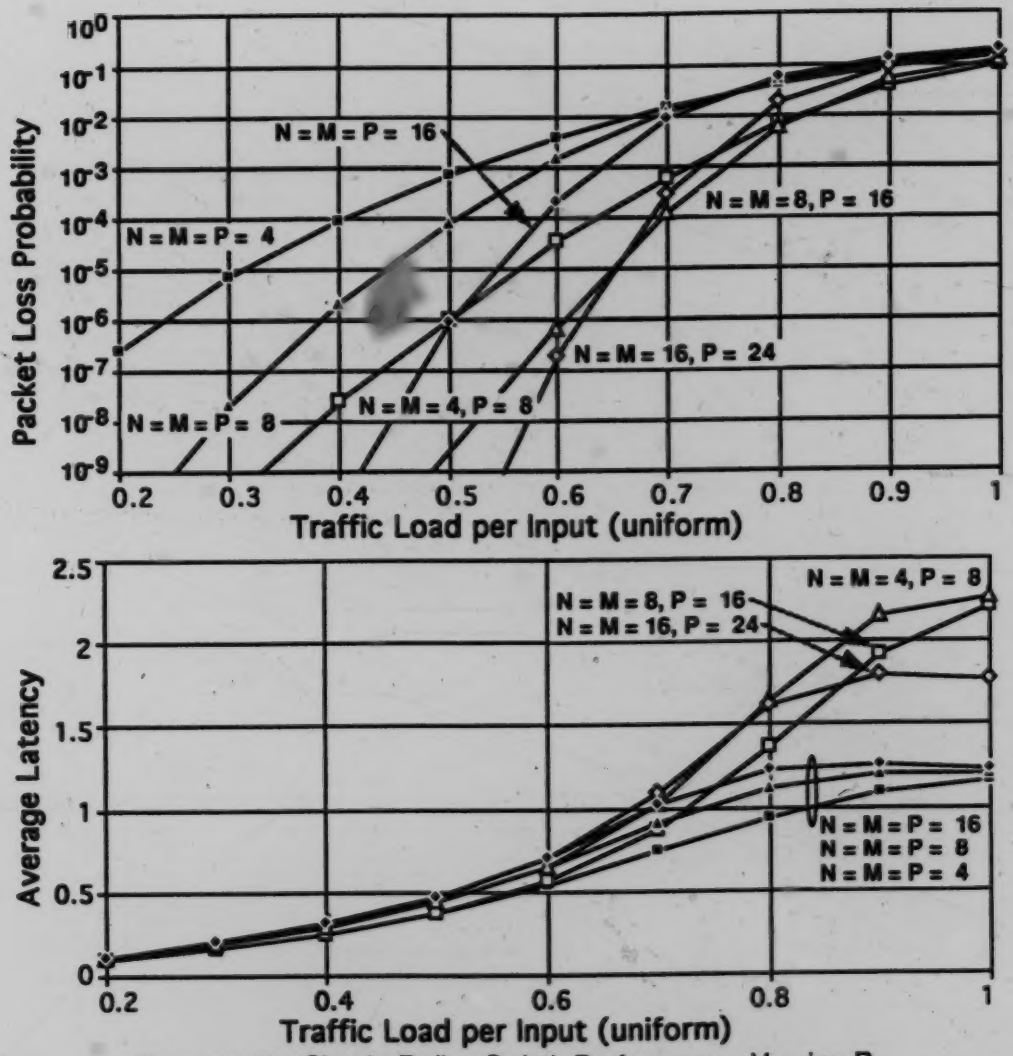


Figure 3.23 - Simple Reflex Switch Performance, Varying P

As noted before, any desired PLP performance level can be achieved by increasing the number of buffers as shown in figure 3.23. Adding buffers has the disadvantage of increasing the average and maximum latency a packet experiences, as well as increasing the number of crosspoints required to implement the switch. The simple reflex switch with the same relative number of buffers has better PLP performance for a larger switch. For example, a square 8 input simple reflex switch with 8 buffers has better PLP performance than a

square 4 input simple reflex switch with 4 buffers. Again, this indicates that this switch architecture will scale to larger sizes in a simple manner if technology allows.

Reflex Switch - Fast Switch Core with Single Packet Buffers

The simple reflex switch uses a very simple control algorithm to maintain packet sequence. Simple controllers take less time to set the crosspoint states than complex controllers, allowing either the number of inlets and outlets to increase at a given bit rate, or the bit rate itself to increase in the simple reflex switch by comparison with the staggered reflex switch. Increasing the number of inlets and outlets may not be feasible because of technology limitations, such as the optical power splitting required or the integration level of the photodetector crosspoints. Although more packets will pass through the switch in a given time, increasing the bit rate of the inputs and outputs will not improve the PLP performance of the switch, since this is set by the number of available buffers. If the switch core and buffer bit rates are increased but the input and output bit rates are left the same, as shown in figure 3.24, the PLP performance of the switch will improve. If the bit rate of the core of the switch is increased by L times, up to L packets can be delivered to a single output in a single output time slot in much the same manner as the knockout switch presented above operates. This results in a hybrid switch architecture that uses shared output buffering (the P fiber delay lines) and simple output buffering on each output (a small electronic FIFO). The fiber delay lines form a bit rate independent RAM buffer, with each fiber delay line holding one packet. Increasing the bit rate of the switch core is not feasible for any switch architecture presented here except the simple reflex switch because of the more complex controller required to maintain packet sequence. This switch architecture will be referred to as the simple fast core reflex switch.

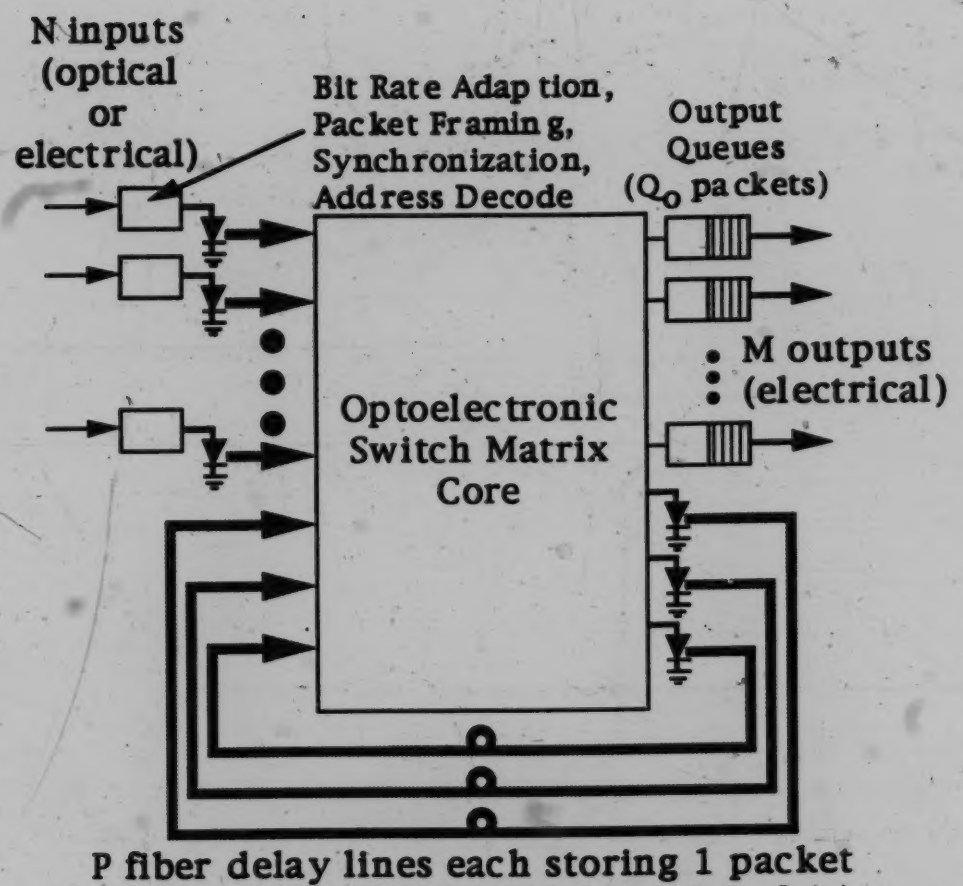


Figure 3.24 - Simple Fast Core Optoelectronic Reflex Switch

As the block diagram in figure 3.24 shows, operating the core of the switch at a different bit rate than the inputs and outputs requires a rate adapter circuit at the input to the switch core that translates between the two bit rates. This operation currently must be done electronically, which is acceptable since the packet framing and address decoding must also be done electronically as discussed earlier. Small queues, each holding Q_o packets, are required at each output to allow more than one packet to arrive at an output in a single output time slot without increasing the PLP of the switch too much. The core speedup, L, can be small compared to that required in the knockout switch because the shared output buffers will buffer the packets that would have been

dropped in the knockout switch. The output queues can also be quite small since only L packets can be delivered to an output in a single time slot and L is small. In addition, a simple back pressure scheme could be implemented to keep a packet in the shared output buffers if the desired output queue is full. This would only slightly complicate the controller as the back pressure scheme could be implemented by indicating that the output was busy even for the highest priority fiber delay line. The results presented here do not use a back pressure scheme.

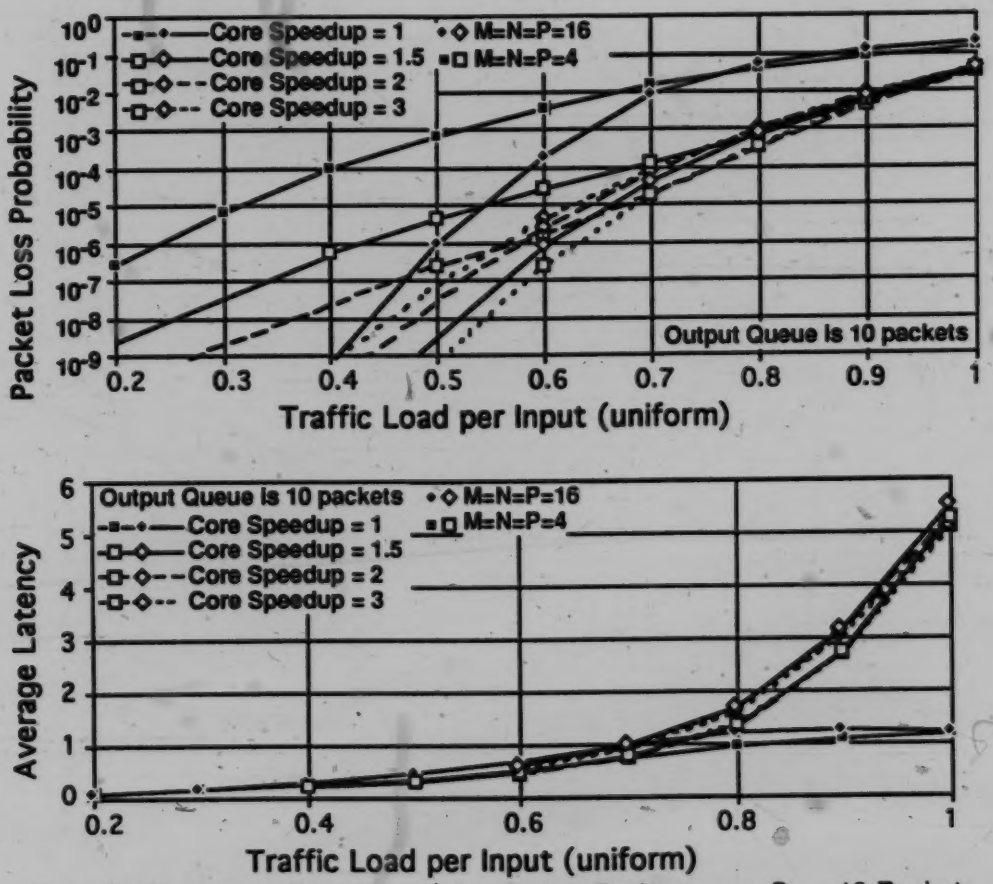


Figure 3.25 - Simple Fast Core Reflex Switch Performance, Q_O = 10 Packets

It is interesting to note that, with only a slight difference, the same two packet loss mechanisms as in the knockout switch are present in this switch

architecture as well. The output queue overflow is the same, but when more than L packets destined for the same output arrive at the switch in a single time slot the knockout switch drops the excess packets, while the reflex switch routes these packets into the recirculating fiber delay lines. When the 16 x 16 switch is used with the 10 packet output queue, the PLP performance is actually degraded by increasing the core speedup from two times to three times. This indicates that this switch is operating in the regime where simple output queue overflow dominates the packet loss compared to packets lost because all the fiber delay lines are occupied. For the 4 x 4 switch, the PLP performance only marginally improves with increasing the core speedup from two to three. This indicates that the smaller switch is also approaching the point where output queue overflow will dominate the packet loss. A back pressure mechanism that keeps packets in the recirculating fiber delay lines if the desired output queue is full should improve the PLP performance of the switch considerably. Another possibility is to increase the queue size as shown below. The average latency performance of the simple fast core reflex switch is almost entirely determined by the latency a packet experiences in the output queue as demonstrated by the similar latency of switches using different core speedup factors.

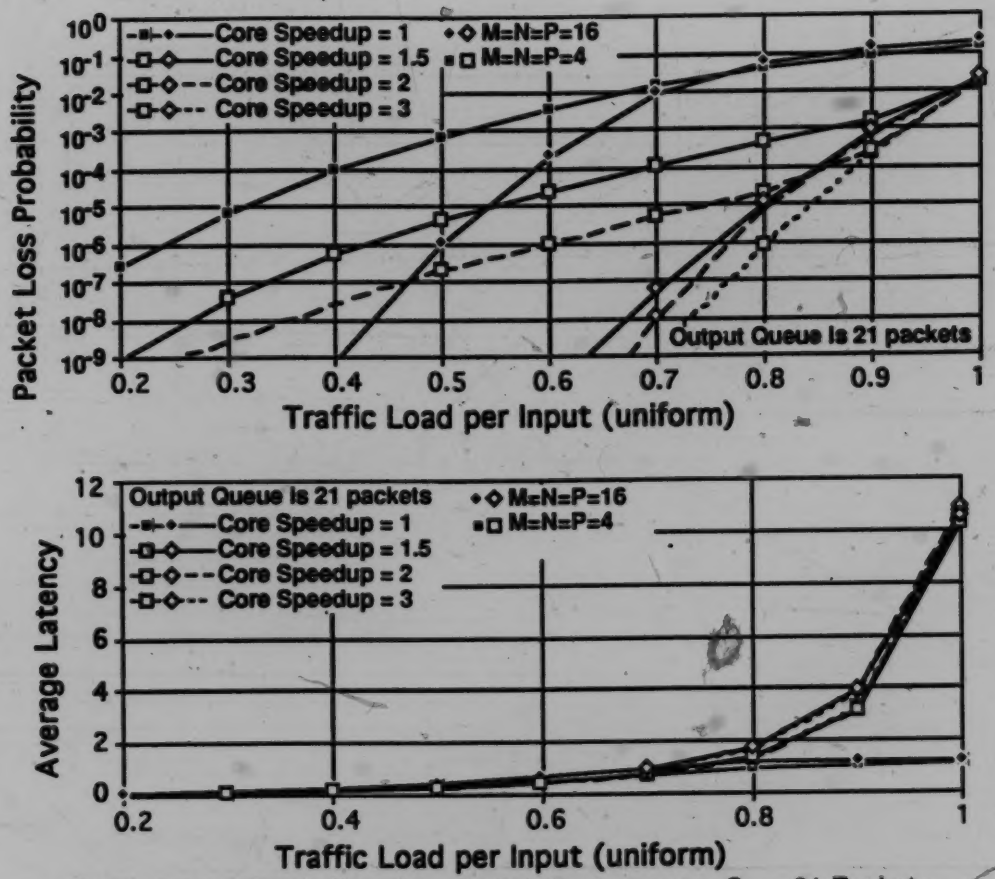


Figure 3.26 - Simple Fast Core Switch Performance, Q_O = 21 Packets

From figure 3.26, the smaller 4 x 4 switch with an output queue size of 21 packets is not operating in the regime where output queue overflow dominates the packet loss. With the larger queue size, the 16 x 16 switch PLP performance only improves slightly with the core speedup factor changed from two to three. Again, this indicates that the packet loss is dominated by the output queue overflow and either a larger queue or a back pressure scheme should be implemented.

	Q_O = 10 Packets	Q_O = 21 Packets	Dominant PLP Mechanism
N=M=4, L=2, ρ=0.6	2×10^{-6}	1×10^{-6}	Packet Collision & Queue Overflow
N=M=4, L=3, ρ=0.6	2×10^{-7}	$< 10^{-9}$	Queue Overflow
N=M=16, L=2, ρ=0.6	4×10^{-6}	$< 10^{-9}$	Queue Overflow
N=M=16, L=3, ρ=0.6	9×10^{-7}	$< 10^{-9}$	Queue Overflow

Table 3.3 - Fast Core Single Packet Buffer Reflex Switch PLP Change with Output Queue Size

Switch Architecture Performance Comparisons

In this section, the switch architectures introduced above are compared based on three criteria, namely, the number of buffer spaces, the number of buffers, and, finally, the number of crosspoints. All of the switch architectures compared in this section have eight inputs and eight outputs. The knockout switch is presented as well to demonstrate the improvement given by adding the shared recirculating fiber delay lines in the simple fast core reflex switch architecture.

First, the various switch architectures are compared based on the number of buffer spaces, each of which holds a single packet, as shown in figure 3.27. The comparisons are based on switches with eight inputs and eight outputs. Instead of eight buffers, the switches will now use 36 buffer spaces (which is the number of buffer spaces in a staggered buffer structure made of eight buffers). The knockout switch curve shows the dramatic PLP performance improvement gained by adding the shared recirculating fiber delay lines.

The staggering switch and the staggered buffer reflex switch have the worst PLP performance, as well as the worst average latency performance under all but the highest traffic loads. The simple fast core reflex switch architecture, operated with the switch core at a bit rate 1.5 times the bit rate of the inputs and outputs and output queues of 21 packets, has PLP performance

better than the staggering switch and staggered buffer reflex switch under high loads but doesn't achieve a PLP of 10^{-9} until a traffic load of about 0.35 which is worse than the staggered buffer reflex switch but better than the staggering switch. The simple fast core reflex architecture operated with a core speedup factor of two has even better PLP performance, reaching a PLP of 10^{-9} at a traffic load of approximately 0.55. The simple fast core reflex architectures shown here actually only use 29 buffer spaces per output, so the PLP performance would improve somewhat if the number of recirculating fiber delay lines were increased to 15 to have 26 buffer spaces available. It is interesting to note that the average latency of the switch architectures using simple output queues is considerably higher at high traffic loads. This is simply due to the greater number of buffer spaces (21 packets at each output) available, resulting in more packets being delayed longer amounts of time when the traffic load is high. The simple reflex architecture, with only 16 buffers and 16 buffer spaces rather than 36, has comparable PLP performance to the simple 2x core reflex switch when operating under traffic loads where the PLP is about 10^{-9}. If the simple reflex switch used all 36 buffer spaces, its PLP performance would be the best of this comparison by a large margin.

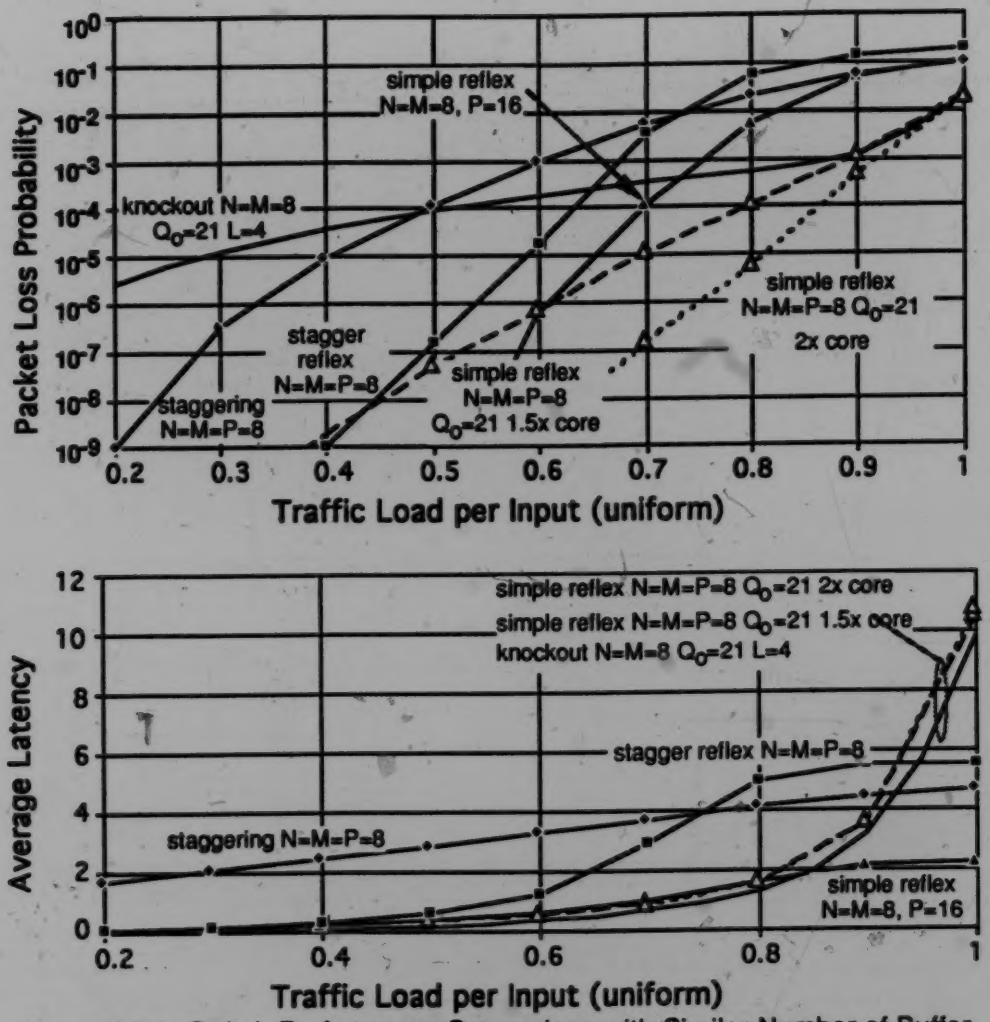

Packet Loss Probability
Traffic Load per Input (uniform)
simple reflex N=M=8, P=16
knockout N=M=8 Q0=21 L=4
staggering N=M=P=8
stagger reflex N=M=P=8
simple reflex N=M=P=8 Q0=21 1.5x core
simple reflex N=M=P=8 Q0=21 2x core
Average Latency
Traffic Load per Input (uniform)
simple reflex N=M=P=8 Q0=21 2x core
simple reflex N=M=P=8 Q0=21 1.5x core
knockout N=M=8 Q0=21 L=4
stagger reflex N=M=P=8
staggering N=M=P=8
simple reflex N=M=8, P=16

Figure 3.27 - Switch Performance Comparison with Similar Number of Buffer Spaces

Unlike the other components used in constructing a switch, a length of fiber capable of storing a single packet at a given bit rate has approximately the same cost as a length of fiber capable of storing more than one packet. A comparison which is more indicative of the overall complexity and cost of the switch architecture is based on the number of crosspoints used. Figure 3.28 compares the various switch architectures, each using 256 crosspoints. This comparison allows the staggering switch to use 16 fiber delay lines for a total of

136 buffer spaces, while the staggered buffer reflex switch uses eight buffers with 36 buffer spaces, and the simple reflex switch architectures use eight buffers with eight buffer spaces plus the output queues. As shown, the simple reflex switch has the worst PLP performance and best average latency performance in this case due to its limited buffer space. The staggered buffer reflex switch has slightly better PLP performance than the simple reflex switch. The simple fast core reflex switch with a speedup factor of 1.5 has similar PLP performance to the staggered buffer reflex switch when operated at a PLP near 10^{-9}. The staggering switch has considerably better PLP performance due to the extra eight buffers, although its average latency performance is still the worst when operated under light loads. The simple 2x fast core reflex switch has PLP performance similar to that of the staggering switch, with both achieving a PLP of 10^{-9} at a traffic load of approximately 0.6. The simple 2x core reflex switch has an average latency about 2.5 time slots lower than that of the staggering switch.

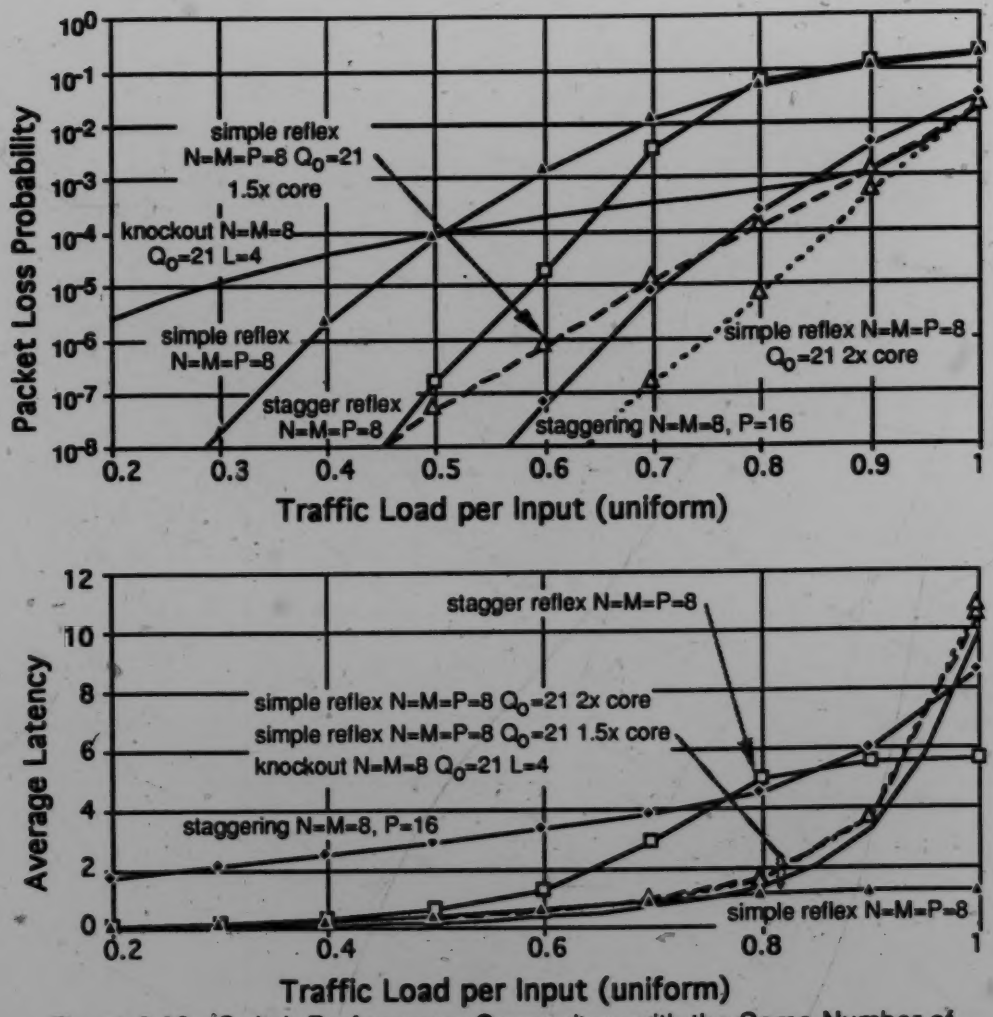

Packet Loss Probability
simple reflex N=M=P=8 Q0=21 1.5x core
knockout N=M=8 Q0=21 L=4
simple reflex N=M=P=8
stagger reflex N=M=P=8
staggering N=M=8, P=16
simple reflex N=M=P=8 Q0=21 2x core
Traffic Load per Input (uniform)
Average Latency
stagger reflex N=M=P=8
simple reflex N=M=P=8 Q0=21 2x core
simple reflex N=M=P=8 Q0=21 1.5x core
knockout N=M=8 Q0=21 L=4
staggering N=M=8, P=16
simple reflex N=M=P=8
Traffic Load per Input (uniform)

Figure 3.28 - Switch Performance Comparison with the Same Number of Crosspoints

When photodetectors are used as crosspoints, the O/E conversion is used as the switching mechanism. This implies an E/O conversion must be done before a signal is sent into one of the fiber delay line buffers. Each photodetector crosspoint can be a simple structure, such as a MSM-PD, and the 'column' of crosspoints attached to a single outlet can be integrated together. This allows relatively inexpensive crosspoints, although the E/O conversion is still expensive. This allows another comparison of switch architectures to be

made based on the number of fiber delay line buffers used, which is a good indicator of the complexity and cost of a receiver-switched optoelectronic switch architecture.

The different switch architectures each with eight fiber delay line buffers are compared in figure 3.29. The staggering switch and the simple reflex switch have similar PLP performance, although the simple reflex switch has much better average latency performance, a simpler controller, and many fewer buffer spaces. The staggered buffer reflex switch has considerably better PLP performance than either the simple reflex switch or the staggering switch, showing a PLP of 10^{-9} at a traffic load of 0.4 compared to approximately 0.25 for the simple reflex switch and 0.2 for the staggering switch. The simple fast core reflex switch has slightly better PLP performance than the staggered buffer reflex switch, even when used with a small output queue holding 10 packets and the switch core operating only 1.5 times as fast as the inputs and outputs. As the curve for the knockout switch with four paths to each output shows, the eight shared recirculating fiber delay lines used in the simple fast core reflex switch architecture improves the PLP performance considerably at the expense of an increase in the average latency of approximately half a time slot.

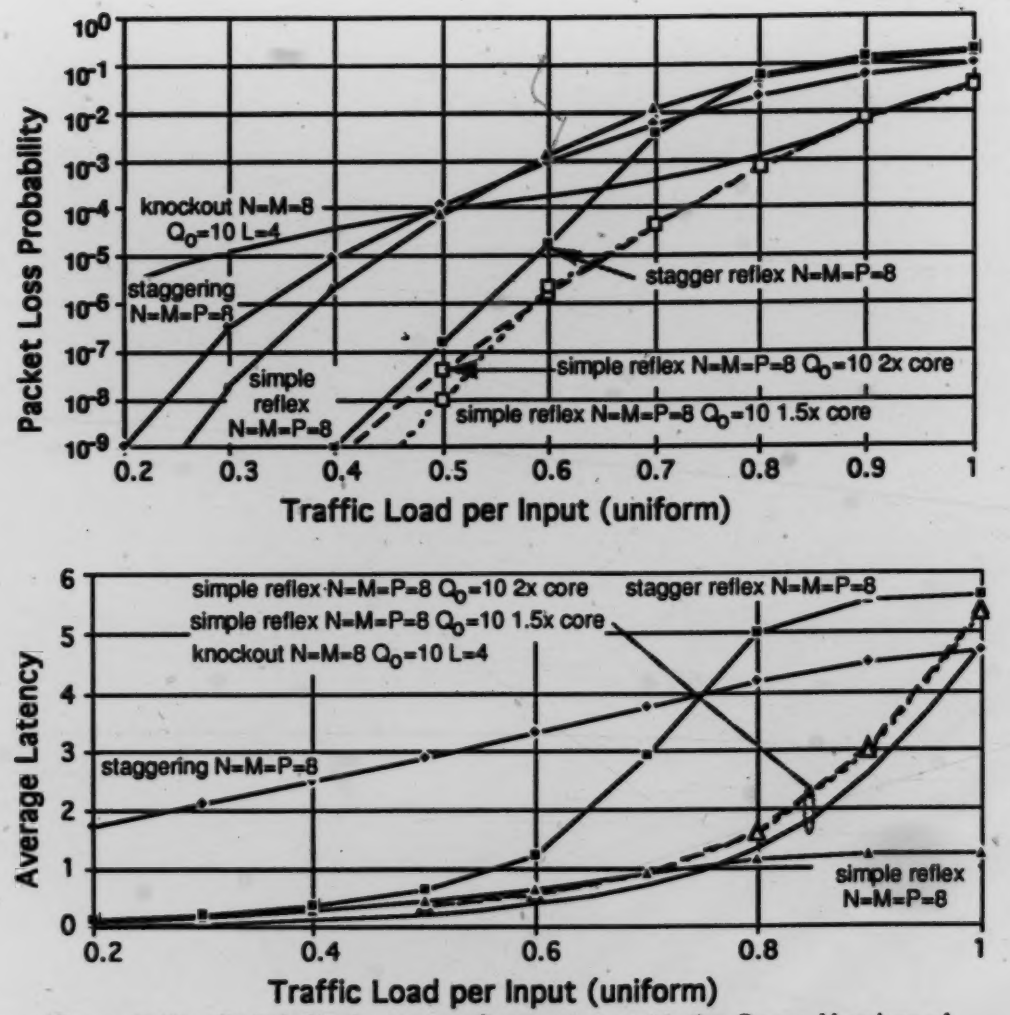


Figure 3.29 - Switch Performance Comparison with the Same Number of Buffers

As these comparisons show, each of the switch architectures has advantages and disadvantages. To decide upon a switch architecture the criteria for comparison must be decided and a priority given to each. Obviously, the PLP performance of a switch is a very important consideration, as it impacts the user considerably through the retransmission of lost data. On the other hand, consider that the largest average latency found in the switch architectures compared above was just less than 11 time slots. At an input and output OC-24

bit rate of 1.244 Gbps, this results in an average delay of 3.8 μsec for a packet passing through the switch. In comparison, a length of fiber 1 km long adds a delay of 4.8 μsec. The latency of the high speed switches considered here is insignificant for most, if not all, applications and should not be a major factor in the selection of a switch architecture. The ability of each of the switch architectures to be scaled to larger sizes is good, assuming that the technology used allows the scaling (for example the loss in the optical power splitting must be low enough to permit large switch matrices). The staggering switch has an advantage in scalability because it requires fewer crosspoints. The simple reflex switch architecture has the advantage of using a simple controller that will maintain packet order, allowing for faster bit rates, or more inputs and outputs, or both, for an equally complex controller.

For a fast packet switch using optoelectronic receiver-switched elements, the number of buffers (since each additional fiber delay line buffer requires an additional laser source) is the most important cost constraint. For this reason, the simple reflex switch architecture was chosen for further study. The individual components that constitute the fast packet switch based on a signal distribution space division multiplexed switch core that uses receiver-switched optoelectronic elements are examined next.

Switch Components

A block diagram of a receiver-switched optoelectronic switch based on the simple reflex architecture is shown in figure 4.1. The components shown include: packet framing and clock recovery; address to output decode; packet synchronization; electronic amplifiers, laser sources; optical signal distribution; photodetector arrays (together forming a crossbar switch core); fiber delay lines; address replacement; simple output queue, and controller. Of these components, several are common to electronic, almost-all optical (optical switches with an electronic controller), and optoelectronic fast packet switches and will be discussed only briefly. These common components providing: packet framing and clock recovery; address to output decode; packet synchronization; and address replacement blocks, are all at the periphery of the switch. These common blocks are used to either pre-process a packet to prepare it for switching, or to post-process a switched packet in preparation for transmitting it further on its way.

The core components of the switch, those that are specific to the switch matrix implementation, will be discussed in more detail with experimental results given where available. These core components were each characterized in turn, with the goal of demonstrating switching among a single array of photodetectors. Unfortunately, this goal was not reached, although most of the components were characterized.

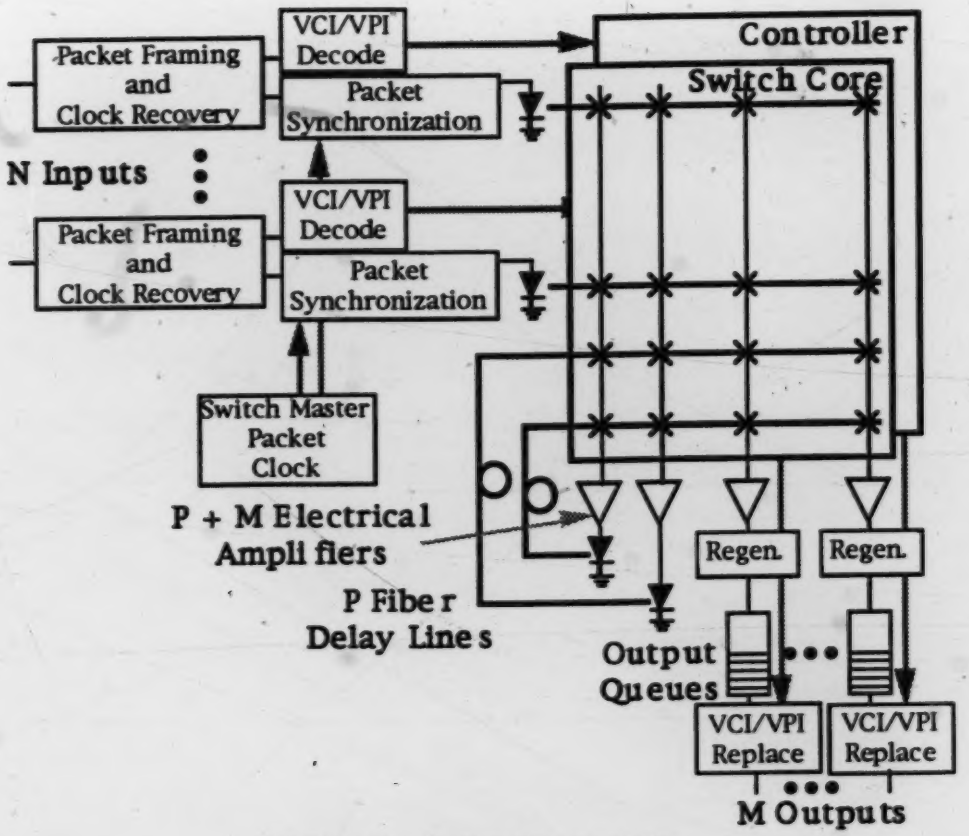


Figure 4.1 - Switch Components

Packet Framing and Clock Recovery

The packet framing and clock recovery circuit is used to recover the start of the packet, as well as the bit and packet clocks, from within the arriving bit stream. All these operations are required for any fast packet switch, and must be performed electronically in the standard manner [Bri94][Ena92] for any currently feasible implementation. Research results have been reported in [Smi92], [Ell93], [Pat94], and [Obr94] on methods to recover the bit clock using all-optical components but all consist of complex schemes that rely upon high optical power levels and the non-linear transfer functions present in certain optical components. The optical framing system presented in [Obr94] are particularly interesting in that it recovers the packet clock as well , although the

method used does not lend itself to ATM switching. This is because the ITU ATM standard [ITU93] mandates the framing on the CRC byte in the header, rather than adding extra bits around the 53 byte cell that are solely used for framing as is done in [Obr94].

The CRC byte of the five byte header of each cell is used for detecting (and possibly correcting) errors in the header and for framing on the ATM cells within the bit stream [Dod93]. The process of framing is needed so the switch knows where the boundaries of an arriving cell are within the bit stream on that inlet. The last eight bits of the header form a CRC of the previous 32 bits in the header based on the generating polynomial shown in equation 4.1.

$$G(x) = x^8 + x^2 + x + 1 \qquad \text{Eq. 4.1}$$

The result of this polynomial will be zero for a valid 40 bit header, and will be non-zero for any other 40 bits from the arriving bit stream. The 8 bits of CRC data used to verify the other 32 bits of the header not only detect and correct all single bit errors, but also detect 89% of the cases [dPr95] where more than one bit in the header is in error. To guard against erroneous header data two states are used in the error detection and correction circuitry. If no errors have been found, the correction state is used which corrects single bit errors and detects multibit errors. When an error is detected, the error is corrected if possible, and the circuitry enters the detection state which only detects errors and doesn't correct any errors. As soon as a valid header is received, the circuitry goes back to the correction state. This is done to reduce the chance of a multibit burst error being considered a correctable single bit error. This block uses standard CRC generation and detection circuitry that is complex enough to require electronic implementation currently. Standard serial implementations can be used, such as [Low93], although parallel implementations [Alb90] have also

been proposed.

Of course, there is always the possibility (1 chance in 2^8 or about 0.25 %) that the data portion of the arriving ATM cell will mimic (intentionally or unintentionally) a valid CRC byte. The solution to the unintentional mimic of the CRC byte is to check for the valid header five byte pattern starting every 53 bytes since idle cells are used in the ATM physical layer [dPr95] to maintain synchronization. As long as the input is in frame (the cell clock is synchronized to the cells within the arriving bit stream), the data portion is not checked for a valid CRC byte since one is already present. When the input is out of frame (searching for the cell within the arriving bit stream), if the CRC fails, then the ATM cell is not correctly framed, so delay the potential 53 byte cell frame one bit and try again. The cell clock is declared in frame when N valid CRCs arrive consecutively. Choosing a high N decreases the probability of improperly declaring a false frame, but increases the time taken to declare a true frame. Going out of frame requires receiving M consecutive invalid CRCs. Choosing a high M decreases the probability of losing frame improperly, but increases the amount of erroneous data transmitted before loss of frame is declared.

To thwart the intentional mimic of the CRC byte in the data portion, the ITU ATM standard uses another generating polynomial ($x^{43} + 1$) as a simple scrambler for the data portion of the ATM cells alone. This scrambler randomizes the data portion of the ATM cell (with each bit being dependent on a bit that occurs more than the size of the header earlier) to prevent the insertion of fake headers by malicious users.

Since the relatively complex framing logic has to be done at the input bit rate (1.244 Gbps for OC-24), which is too fast for normal TTL or CMOS logic, GaAs, or perhaps ECL, can be used. If dummy bits between cells (discussed below) are used to allow more time for the signal switching, the framing can be

performed on the specific pattern of these dummy bits. This may reduce the complexity of the required logic enough to allow the use of less expensive CMOS logic.

Address to Output Decode

The address to output decode block is simply used to convert the destination address of the packet into its desired output of this switch. In the simple case of a self-routing switch, where each bit of the destination address represents the routing choice at a single 2 x 2 switching element, this function is not required. All other switch architectures require some method to map the destination address to an output of the switch. The usual solution is to have an electronic associative memory that uses the destination address as the key, and produces the desired output as the result. The associative memory has an entry added or deleted whenever a call is established or terminated. Since this block only operates at the packet rate (not the bit rate) of an input, CMOS logic and memories [Qui94] can meet the performance requirements.

In addition, ATM switching presents the special problem of allowing the reuse of VCI/VPI identifiers on the different links. This requires the replacement of the VCI/VPI on the outgoing cell, forcing another associative memory at each input to return the VCI/VPI pair to be used on the output link as well. In practice, the single associative memory at the input is expanded to return both the desired output and the VCI/VPI to be used at the output.

Packet Synchronization

Another problem is the synchronization of the incoming packets on each of the inlets to the 'master' packet clock used within the switch (so the reconfiguration of the switch occurs between cells and not in the middle of cells). This requires either high speed buffering of at least one packet, or wasted bandwidth in the incoming signal so that the switching can be

performed without exact synchronization [Bor93]. Again, this problem occurs whether the switch uses optical, optoelectronic, or electronic switching elements. In current switches, electronic buffering of the signal is used which also allows the insertion of dummy bits between the cells for easier framing logic and slower switching speeds, as well as the possibility of sending the data in a coded format (such as duobinary) to reduce the bandwidth requirements of the detectors and receivers (at the expense of sensitivity).

An electronic FIFO usually forms the buffer, although switching to various fiber delay line lengths [Bal94][Haa92a] has been attempted. Using switched fiber delay lines alone to synchronize an input requires a large number of delay lines, varying in length from less than one bit to one half of a packet duration. Chaining some or all the fiber delay lines together allows delays longer than the longest individual delay line. Another method for realizing a variable fiber delay line uses a piezo-electric transducer to physically stretch the fiber [The88] to increase its delay. This method results in approximately 2% delay variation limited by the elasticity of the fiber, which is too small a delay change for this application.

Another possible method would heat or cool a length of fiber attached to each input to synchronize the arriving packets. This method can not be used to change delays quickly, but packet synchronization does not require quick delay changes. A small synchronization FIFO of a few bits could be added to allow small delay variations quickly, with the temperature being adjusted to compensate on a longer time scale. Calculations for this synchronization scheme will be presented in the section on fiber delay lines, demonstrating the impractical fiber lengths and/or temperature swings required.

Ideally, synchronizing the packets sent through the fiber delay lines in the switch core is not required since the packets are synchronized upon entering

the switch and should therefore stay synchronized throughout the switch. In the real world, with, for example, temperature effects, a fast memory of a few bits would be added between the electrical outlet of the switch and the laser driving the fiber delay line inlet for those inlets and outlets used with the fiber delay lines. The data bits would be clocked out of this small FIFO so that the optical data stream emerging from the other end of the delay line would be synchronized to the master clock used in the switch. Only a few bits are required for this purpose because the packets exiting at the outlets are all synchronized to the master clock and the fiber delay lines have approximately one packet delay.

Electronic Amplifiers

Full modulation of a typical diode laser source requires a 20 mA current (10 dBm) RF electrical signal. Full modulation of the source is desirable since the data signal may traverse the fiber delay lines several times, undergoing an O/E and an E/O conversion on each pass. Given an average laser source optical power of 5 mW passing through 15 dB of losses (the 1 to 10 splitter and two splices) gives -8 dBm of optical power incident upon the switching photodetectors. Typical MSM photodetectors have a responsivity of approximately 0.2 A/W, resulting in a photocurrent of 31.6 μA. Assuming the standard 50 ohm connection, this results in an input electrical power of 25nW or -46 dBm to the electronic amplifier. Therefore, the electronic amplifiers require a gain of at least 56 dB to allow the output of a fiber delay line to fully modulate the attached laser source.

For this large gain, three amplifier stages were used in the experiments. The two initial stages, both within the RF amplifier, each used a Avantek INA-03179 silicon bipolar MMIC amplifier with the schematic shown in figure 4.2 on a single substrate housed in an machined aluminum case. This amplifier circuit

was designed by David Clegg, with myself and Rohit Sharma performing the assembly and testing. The details of the design and assembly are in [TRL94a]. Eleven of these amplifiers were assembled and tested, with the typical results summarized in table 4.1. All of these amplifiers performed consistently and to expectations.

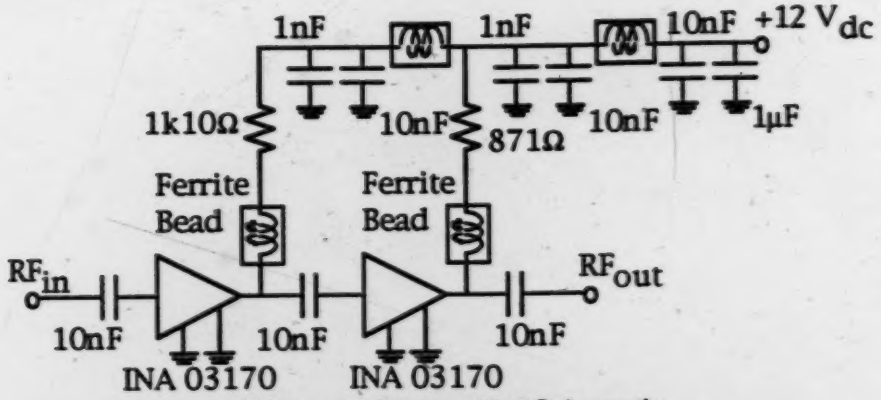


Figure 4.2 - RF Amplifier Schematic

RF Amplifier	Supply Current (mA)	Avg. Gain (dB)	Worst Peak (dB)	Freq. Low -3 dB (Hz)	Freq. High -3 dB (Hz)	NF 0.7 GHz (dB)	NF 1 GHz (dB)	NF 1.5 GHz (dB)
9402001	18.33	49.33	1.33	3.00E+05	2.22E+09	2.99	2.94	3.24
9402002	18.13	49.95	0.80	3.00E+05	2.20E+09	2.93	3.19	3.48
9402003	18.18	50.06	1.90	3.03E+05	2.19E+09	3.08	3.20	3.85
9402004	18.00	49.85	1.82	3.00E+05	2.21E+09	2.65	3.15	3.15
9402005	18.14	49.59	1.58	3.00E+05	2.20E+09	2.31	2.79	3.11
9402006	18.30	48.95	0.90	3.00E+05	2.13E+09	3.12	3.19	3.40
9402007	18.27	49.05	0.71	3.00E+05	2.12E+09	2.66	3.06	2.98
9402008	18.48	48.79	0.65	3.00E+05	2.13E+09	2.64	2.81	2.92
9402009	18.07	49.83	1.52	3.00E+05	2.22E+09	2.55	3.14	3.51
9402010	18.15	49.97	1.82	3.00E+05	2.24E+09	2.53	2.84	2.92
9402011	17.97	49.99	1.00	3.00E+05	2.26E+09	2.54	2.98	3.44
avg.	18.18	49.58	1.28	3.00E+05	2.19E+09	2.73	3.03	3.27
best	17.97	50.06	0.65	3.00E+05	2.12E+09	2.31	2.79	2.92
worst	18.48	48.79	1.90	3.03E+05	2.26E+09	3.12	3.20	3.85
std dev	0.16	0.47	0.48	8.44E+02	4.68E+07	0.26	0.16	0.29

Table 4.1 - RF Amplifier Measured Performance

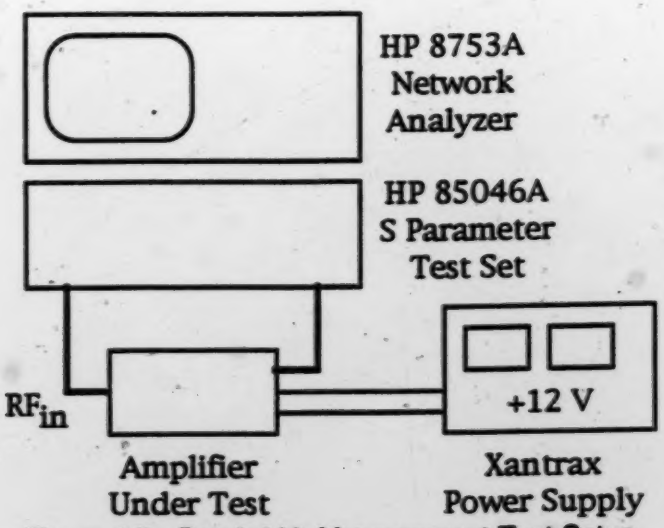


Figure 4.3 - Bandwidth Measurement Test Setup

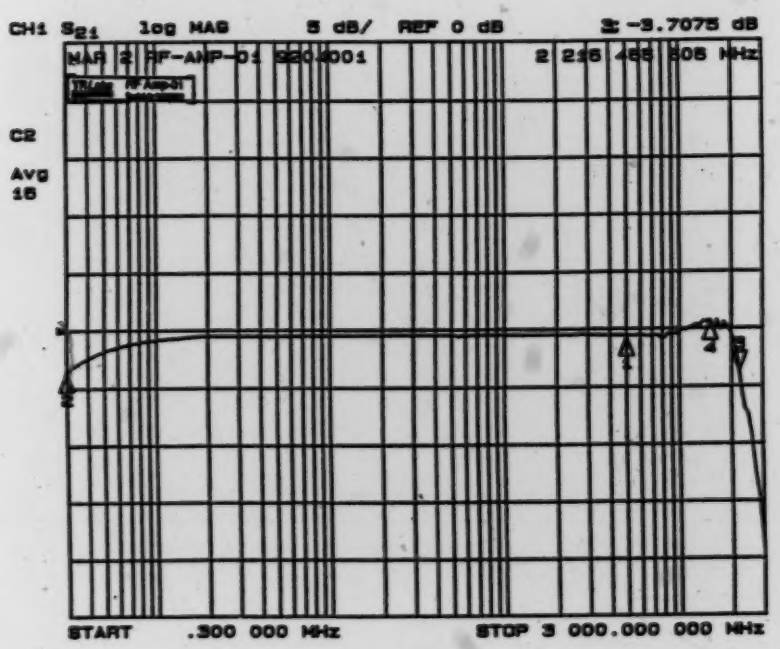


Figure 4.4 - RF Amplifier Measured Bandwidth

A typical bandwidth measurement is shown in figure 4.4, using the test setup of figure 4.3. An HP 8753A network analyzer with a HP 85046A S-

parameter test set was used for the amplifier bandwidth measurements. The power level input to the amplifier under test was -70 dBm. Noise measurements were made using an HP 7000A spectrum analyzer as shown in figure 4.5 with a 50 ohm terminator on the amplifier input.

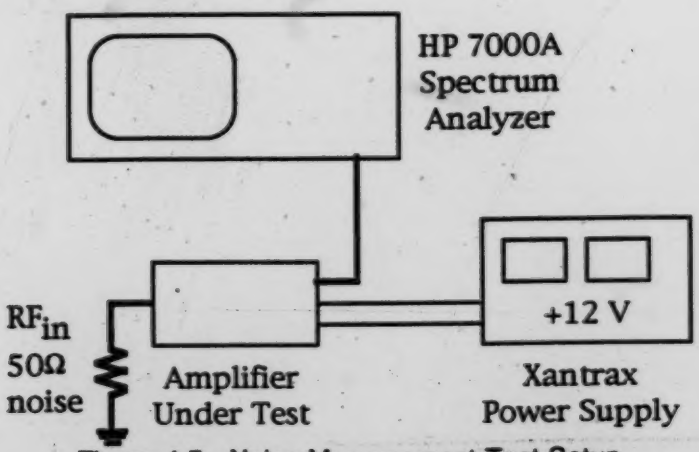


Figure 4.5 - Noise Measurement Test Setup

The spectrum analyzer noise floor is shown in figure 4.6, with a typical amplifier's measured noise in figure 4.7. From these results, the noise levels of the spectrum analyzer, the amplifier under test and the spectrum analyzer, and finally, the noise level of the amplifier under test alone can be calculated as shown in Appendix D. The spectrum analyzer noise figure was calculated as 45.2 dB at 1 GHz. The noise figure of the entire system (amplifier and spectrum analyzer) is calculated as 3.6 dB, resulting in a noise figure for the RF amplifier alone of 2.9 dB.

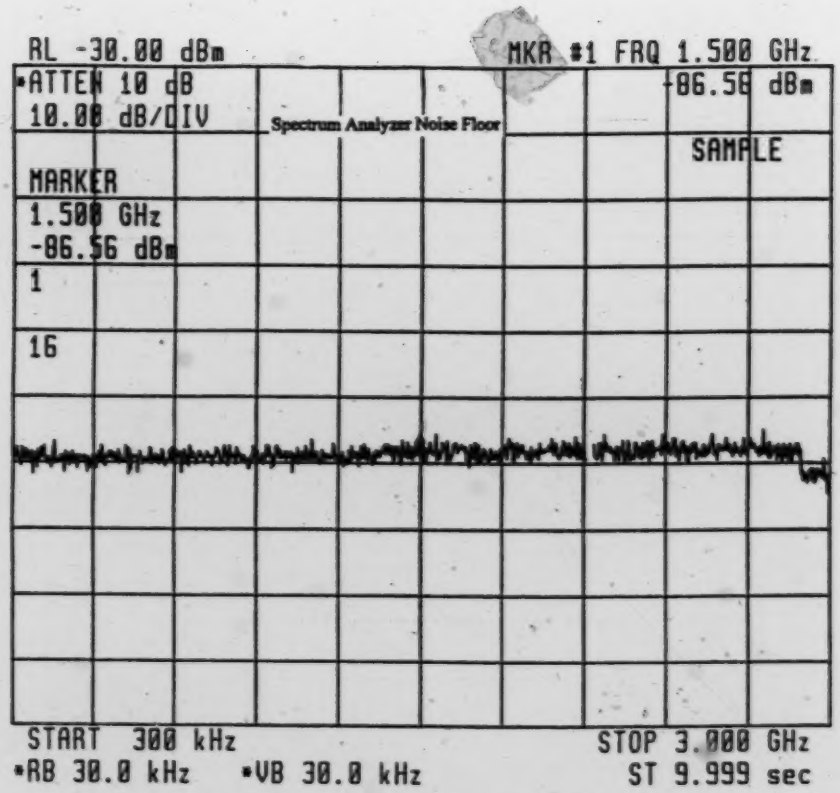


Figure 4.6 - Spectrum Analyzer Measured Noise Floor

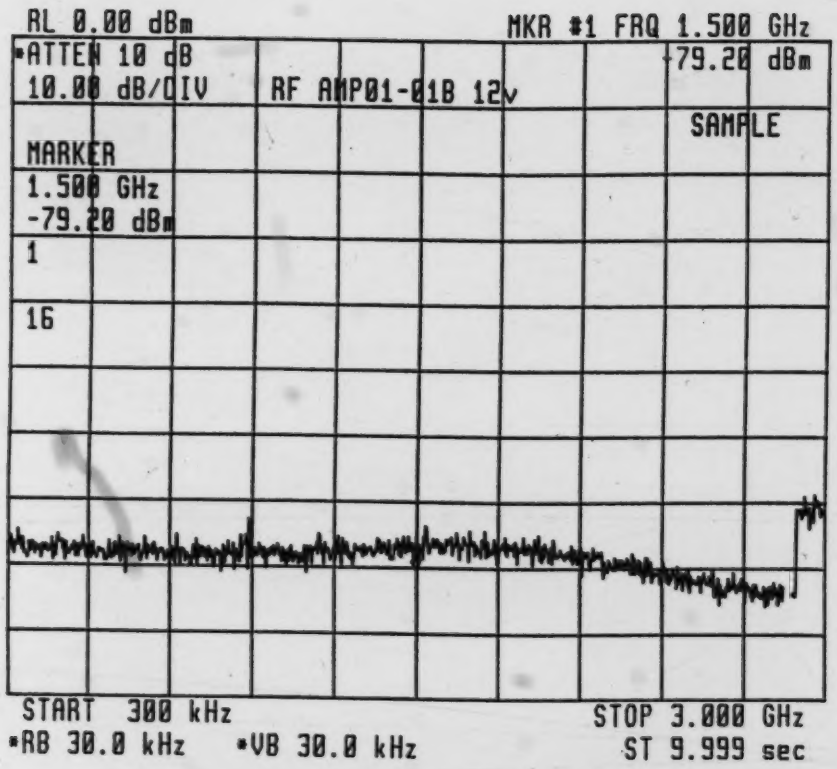


Figure 4.7 - RF Amplifier Measured Noise

The final stage of amplification, the power amplifier, used a single Avantek MSA-0435 MODAMP silicon bipolar MMIC amplifier that allows a higher output power (12.5 dBm 1 dB compression point typically rather than 1 dBm). This amplifier circuit, a slightly modified version of the RF amplifier, was designed by myself and Rohit Sharma [TRL94b] and is shown in figure 4.8. This amplifier was built using the same substrate and aluminum housing as the RF amplifier, although the substrate had the 50 Ω strip line connected across the second stage using a zero ohm surface mount resistor. Ten of these amplifiers were assembled and tested, with the results shown in table 4.2. The only problem noticed with the power amplifiers was that the MMIC amplifier ground leads were not truly at ground. This defect could be improved by adding several more grounding vias to the substrate near the MMIC amplifier ground leads to reduce the inductance. This problem was not noticed on the RF

amplifier due to the lower currents involved. In spite of this small problem, all the power amplifiers performed almost identically. The 3 dB high and low bandwidth limits for this amplifier are actually better than shown but the limits of the HP 8753A network analyzer used to characterize the amplifier was reached.

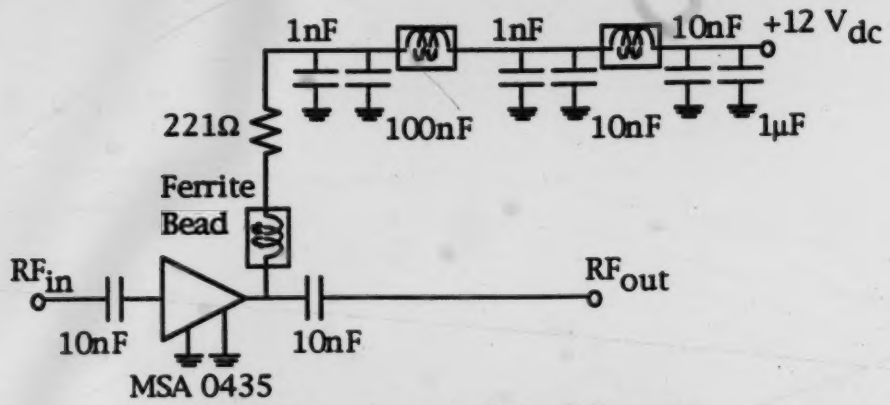


Figure 4.8 - Power Amplifier Schematic

Power Amplifier	Supply Current (mA)	Gain 1 GHz (dB)	Freq. Low -3 dB (Hz)	Freq. High -3 dB (Hz)	NF 0.7 GHz (dB)	NF 1 GHz (dB)	NF 1.5 GHz (dB)
9402001	31.72	6.34	3.00E+05	3.00E+09	6.46	6.90	6.56
9402002	32.53	6.47	3.00E+05	3.00E+09	6.37	7.09	6.97
9402003	31.96	6.53	3.00E+05	3.00E+09	6.62	6.93	6.56
9402004	32.36	6.70	3.00E+05	3.00E+09	6.33	6.42	6.56
9402005	31.73	6.33	3.00E+05	3.00E+09	5.99	6.52	6.22
9402006	32.41	6.66	3.00E+05	3.00E+09	6.55	6.41	6.00
9402007	32.41	6.56	3.00E+05	3.00E+09	6.18	6.71	6.62
9402008	31.79	6.56	3.00E+05	3.00E+09	6.12	6.36	6.87
9402009	32.90	6.59	3.00E+05	3.00E+09	6.40	6.54	6.58
9402010	31.96	6.55	3.00E+05	3.00E+09	6.28	6.35	6.43
avg.	32.18	6.53	3.00E+05	3.00E+09	6.33	6.62	6.54
best	31.72	6.70	3.00E+05	3.00E+09	5.99	6.35	6.00
worst	32.90	6.33	3.00E+05	3.00E+09	6.62	7.09	6.97
std dev	0.40	0.12	0.00E+00	0.00E+00	0.20	0.27	0.28

Table 4.2 - Power Amplifier Measured Performance

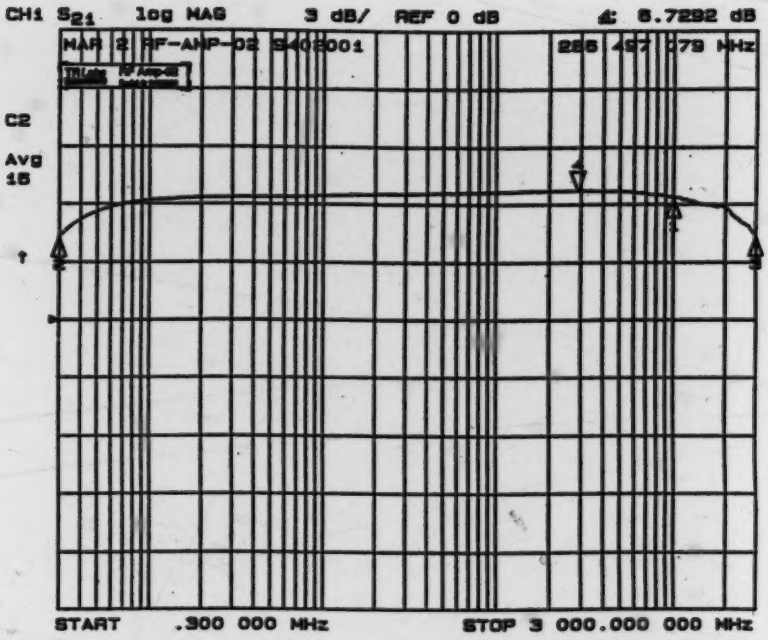


Figure 4.9 - Power Amplifier Measured Bandwidth

Because of the low gain of the power amplifier, an accurate direct noise measurement of this amplifier was not possible. As described in Appendix D, the noise level of the amplifier can be calculated by measuring the combined noise of the RF and power amplifiers together, and then calculating out the known noise level of the RF amplifier and spectrum analyzer to arrive at the noise figures in the summary table.

Finally, the amplifiers were paired up and tested as units (an RF amplifier connected to a power amplifier). No problems were encountered with connecting the two amplifiers together, with a typical result shown in figure 4.10. As this figure shows, the average gain of the amplifier pair is approximately 57 dB at 200 MHz. The maximum gain is 58 dB, the minimum 55 dB, with 3 dB bandwidth limits of 650 MHz and 2.11 GHz.

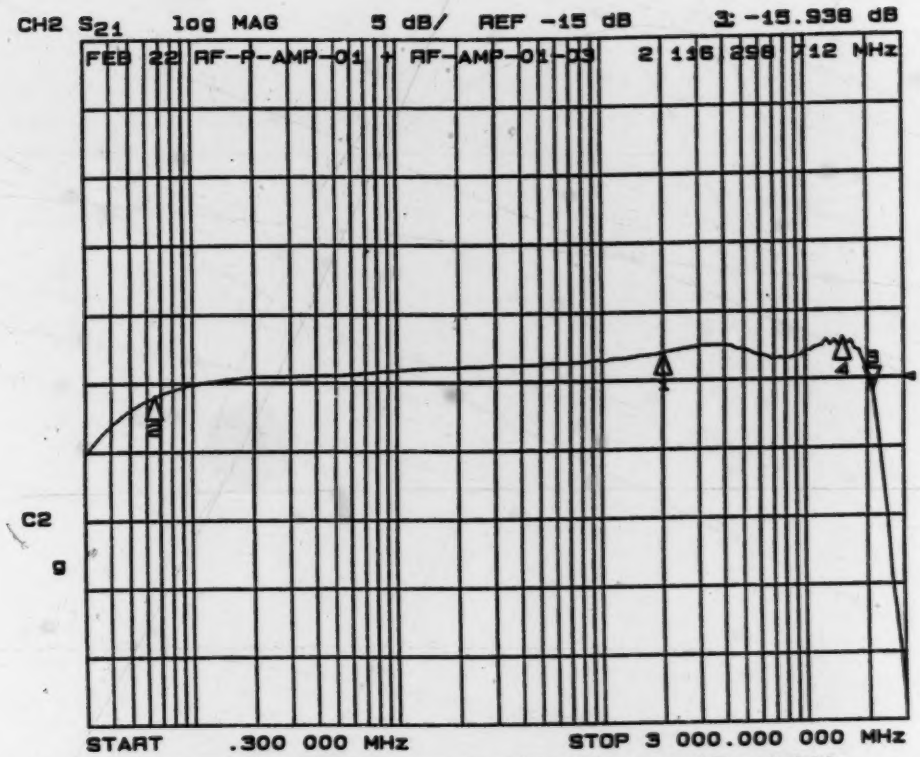


Figure 4.10 - RF & Power Amplifier Measured Bandwidth

In a similar manner to the RF amplifier, the noise level of the combined amplifiers was measured, with a typical result presented in figure 4.11. As detailed in Appendix D, from this measurement the noise level of the entire system can be calculated. The noise level of the spectrum analyzer and the RF amplifier have already been measured and calculated, so they can be removed, leaving only the noise added by the power amplifier. This results in a calculated noise figure of 7.2 dB at 1 GHz for the RF and power amplifiers combined, with a 6.9 dB noise figure attributable to the power amplifier alone.

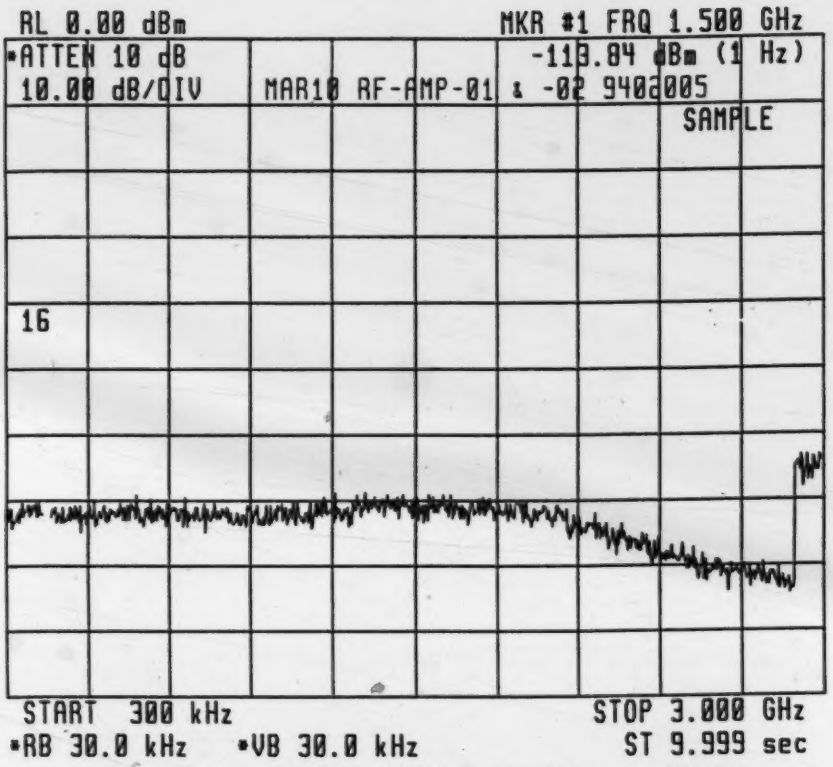


Figure 4.11 - RF & Power Amplifier Measured Noise

Laser Sources

The laser sources are used to generate the optical signals sent into the switch core. A laser source is required at each of the inlets if the fast core, simple reflex switch architecture is being used, but may not be required on the inputs for other switch architectures where the input and core bit rates are the same. If electronic buffering is being used to synchronize the arriving packets, a laser source is required at the inputs as well.

An ideal laser source would have unlimited bandwidth, infinite optical power output, have zero added noise, be perfectly linear, and be economically available. Since ideal lasers are unobtainable, some tradeoffs must be made as detailed exhaustively by M. Veilleux in [Vei90].

For a fast packet switch, the losses in the fiber transporting the optical signal are negligible in comparison to the insertion, distribution, and extraction

losses due to the short fiber lengths involved. The dispersion of the fiber is negligible for the same reason. For these reasons, the lasers are not restricted to the lower loss and dispersion wavelengths of 1300 nm or 1550 nm usually used in high speed communication applications. Standard GaAs lasers, such as those used in compact disc players, are easily obtainable that operate at the 800 nm wavelength region with reasonably high output optical power levels. The Seastar Optics PT150L780 laser was chosen. This package contains a Mitsubishi ML64C11-01 laser diode internally as well as a monitor photodiode. This laser has a nominal wavelength of 780 nm, a threshold current of 40 mA, and a maximum optical power output of 10 mW with a drive current of approximately 75 mA.

The other characteristics of the ideal laser mentioned above are dependent on the circuit used to drive the laser. These characteristics will be discussed after an explanation of the drive circuitry.

Lasers require two drive circuits, a DC circuit that biases the laser above its threshold current (for binary signals, just above the threshold), and an RF circuit that uses the high frequency data signal to drive the laser to a high optical power output when a binary one is being transmitted as shown in figure 4.12. The DC drive circuit used was designed by D. Clegg of TRLabs and has been used in modified forms for several projects [Lam94]. This complex circuit controls the DC bias of the laser, the thermoelectric cooler, the maximum current limit allowed for the laser drive, power on protection, and laser drive and monitor photodiode current display.

The RF drive circuit was also designed by D. Clegg and is a small circuit board that the laser mounts onto and the DC drive circuit connects to. This circuit is designed to receive a 50 ohm RF data signal, capacitively block the DC portion of the data signal, and impedance match the 50 ohm signal to the laser

drive lead. The DC bias signal (from the DC circuit) must also be connected to the same laser drive lead as well, so the DC circuit must present a high impedance to the RF data signal. This circuit's complete schematic and layout are detailed in [TRL94c], while the relevant RF parts are shown in figure 4.12.

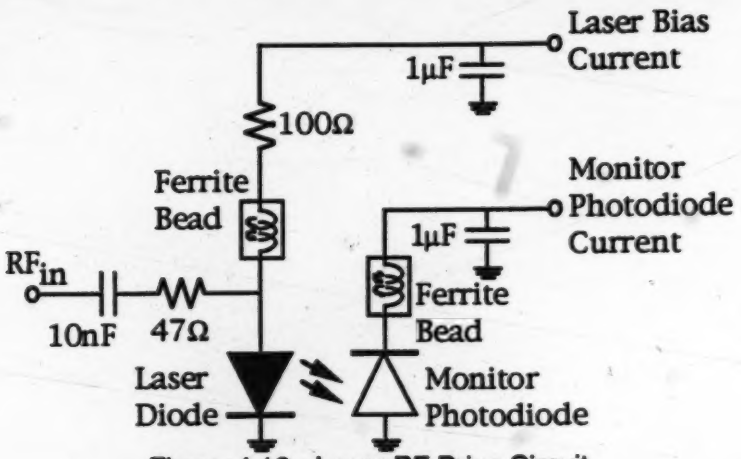


Figure 4.12 - Laser RF Drive Circuit

Ideally, the RF portion of the laser drive circuit would have no bandwidth limits and a constant impedance. Obviously, this ideal cannot be achieved, and the desired bandwidth will affect the design. For this component, a bandwidth of at least 1.5 GHz is desired. Measurements taken on the laser source with the experimental setup shown in figure 4.13 showed the 3 dB bandwidth to be markedly less than expected (less than 800 MHz instead of the desired 1.5 GHz). These measurements were repeated with several different detectors, ranging from MSM photodetectors measured to have greater than 2 GHz bandwidth with a different laser source, to a commercial New Focus wide band detector with a rated bandwidth in excess of 10 GHz. A typical result is shown in figure 4.14, with the test setup used to obtain the results shown in figure 4.13.

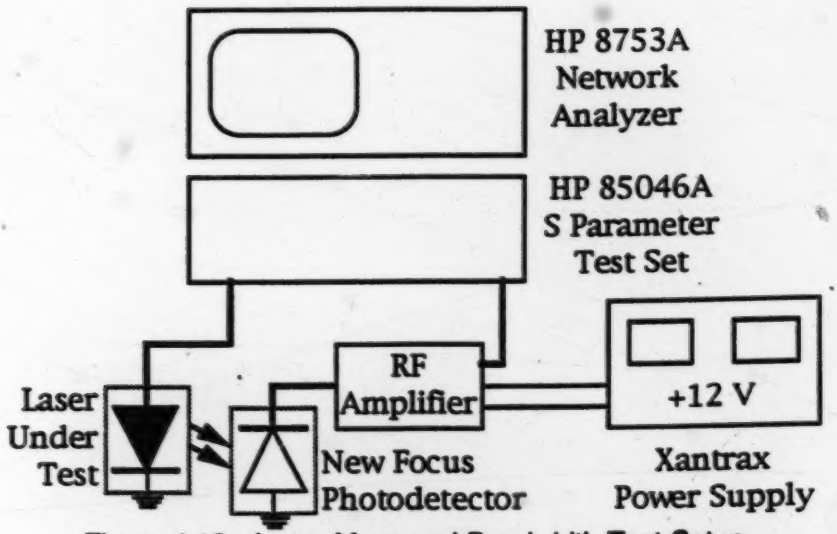


Figure 4.13 - Laser Measured Bandwidth Test Setup

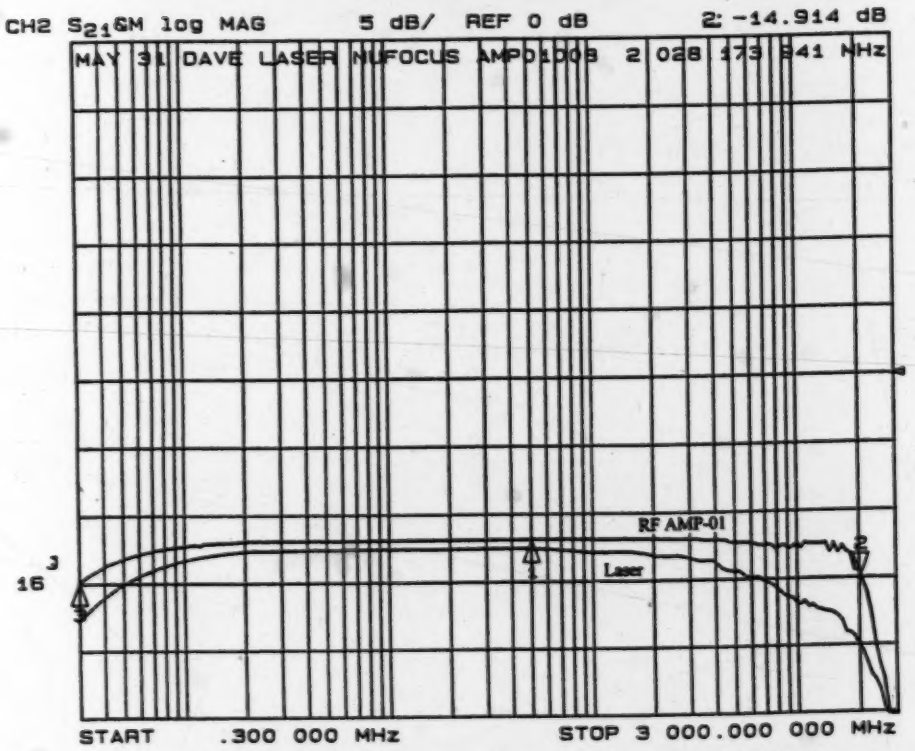

CH2 S21&M log MAG 5 dB/ REF 0 dB 2: -14.914 dB
MAY 31 DAVE LASER NUFOCUS AMP01008 2 028 173 941 MHz
RF AMP-01
Laser
16
START .300 000 MHz
STOP 3 000.000 000 MHz

Figure 4.14 - Laser Measured Bandwidth - No Equalization

Further measurements showed that the laser ground was connected through the ground lead on the laser package which was adding inductance in

the ground path. Actually disconnecting the ground lead on the laser package (leaving the package itself as the ground) produced slightly better results as shown in figure 4.15. The two lines on this figure are for an optical power output of 5 mW and 9 mW from the laser. However, the grounding of the laser is still a problem.

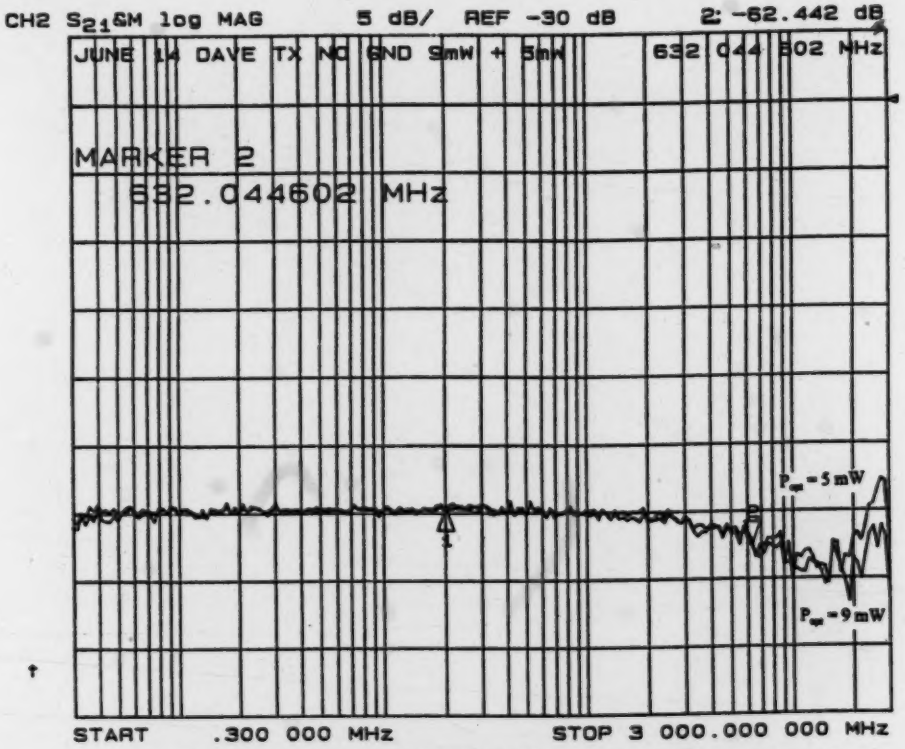

CH2 S21 SM log MAG
5 dB/ REF -30 dB
2: -62.442 dB
JUNE 14 DAVE TX NO GND 9mW + 5mW
632 044 602 MHz
MARKER 2
632.044602 MHz
P = 5 mW
P = 9 mW
START .300 000 MHz
STOP 3 000.000 000 MHz

Figure 4.15 - Laser Bandwidth - Varying $P_{optical}$

Simulations of the RF bias circuit were performed by R. Sharma attempting to match the measured S_{11} response shown in figure 4.16. Assuming reasonable stray capacitance values, these simulations indicated that the inductance of the laser ground lead had to be much greater than the 3 nH specified, with an approximate value of 5 nH. For a similar laser [Lam94] measured an inductance of approximately 6 nH as well.

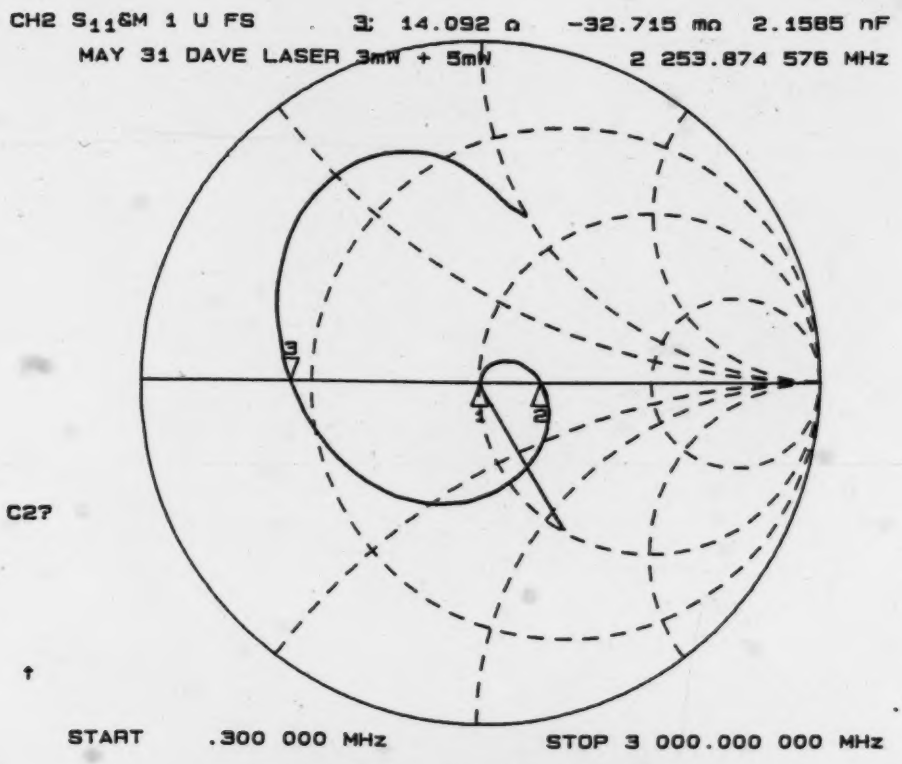


Figure 4.16 - Laser S_{11} Impedance

To achieve the desired bandwidth of 1.5 GHZ, the laser RF bias circuit had to be modified to use equalization. The RF circuit used is shown in figure 4.17, with three components (two capacitors, C_{e1} and C_{e2}, and one inductor, L_e) added. Simulations indicated that C_{e1} should be greater than 1 µF and L_e should be approximately 15 nH. After much experimentation, the best equalized response using this simple equalization circuit was found by actually using a 1 µF tantalum capacitor as C_{e1}, 20 nH inductor for L_e, and a 2.2 pF ceramic capacitor as C_{e2} and is shown in figure 4.18. Obviously, this response was not good enough and a slightly more complex equalization circuit had to be used.

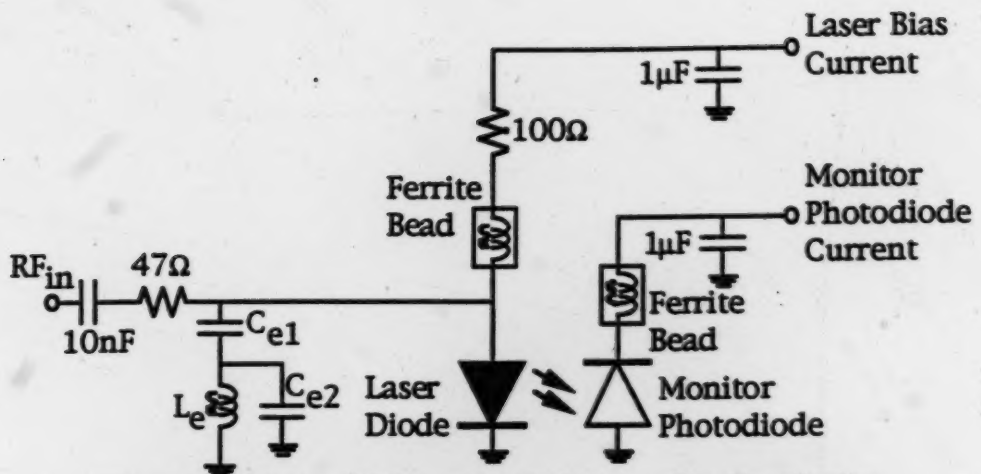


Figure 4.17 - Laser RF Circuit - with C_{e1}, L_e, C_{e2} Equalization

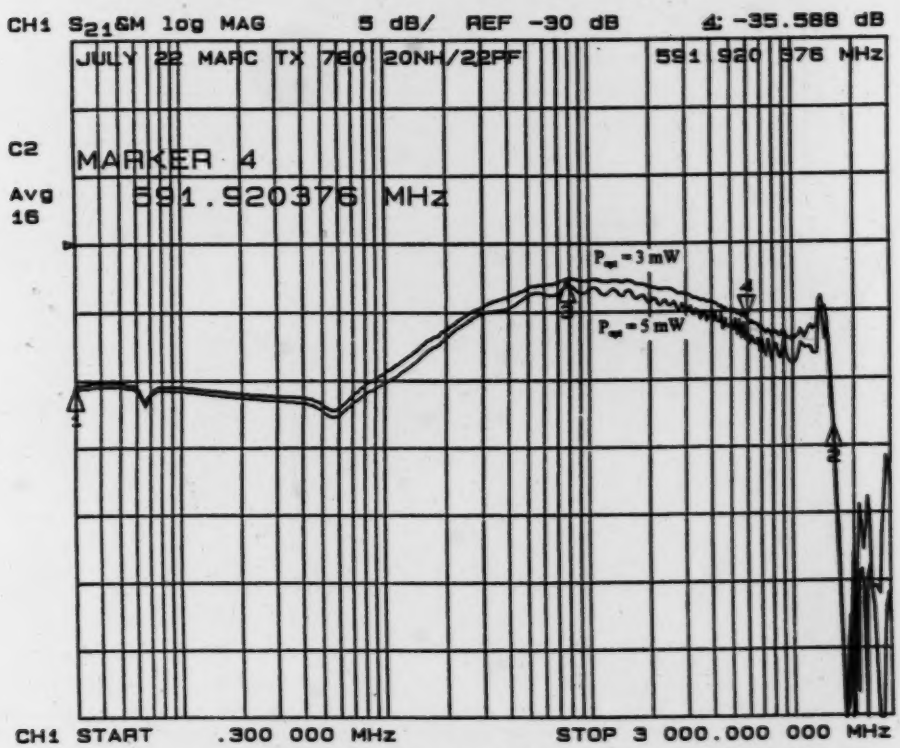

CH1 S21 GM log MAG 5 dB/ REF -30 dB 4: -35.588 dB
JULY 22 MARC TX 780 20NH/22PF 591 920 376 NHz
C2
MARKER 4
Avg 16
591.920376 MHz
Pin = 3 mW
Pin = 5 mW
CH1 START .300 000 MHz STOP 3 000.000 000 MHz

Figure 4.18 - Laser Bandwidth - C_{e1}, L_e, C_{e2} Equalization

To reduce the 100 MHz hump in the response, a resistor (R_e in figure 4.19) was added between C_{e1} and L_e to reduce the resonance of the LC circuit. Again, after much experimentation, the best frequency response was

found using the circuit shown in figure 4.19. The frequency response is shown in figure 4.20.

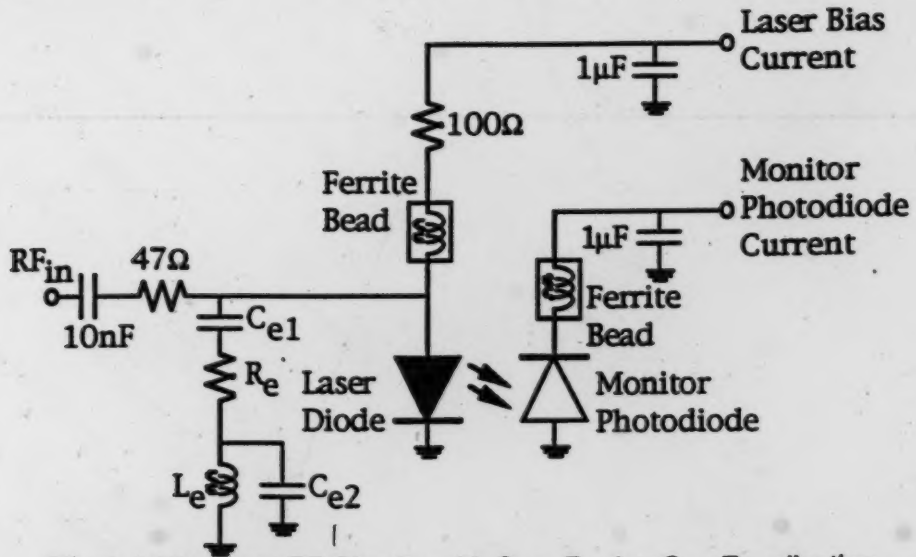


Figure 4.19 - Laser RF Circuit - with C_{e1}, R_e, L_e, C_{e2} Equalization

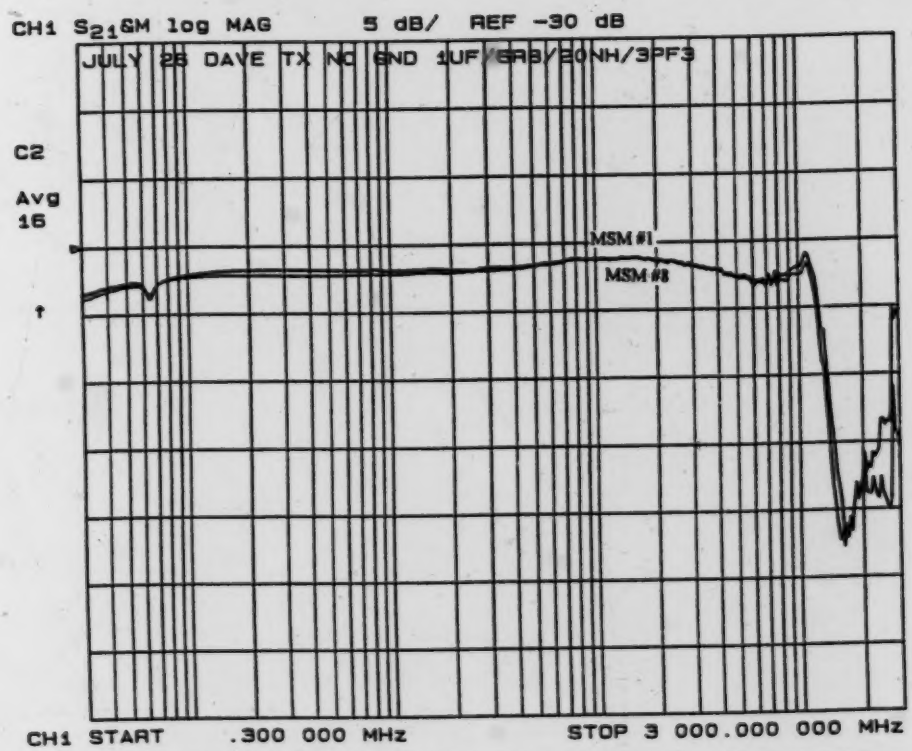


Figure 4.20 - Laser Bandwidth - C_{e1}, R_e, L_e, C_{e2} Equalization

Optical Signal Distribution

An essential part of the receiver-switched optoelectronic switch core is the optical signal distribution. Currently, fiber optic power splitters (1 to 10) are used in the TRLabs prototype switch to split the optical signal from one laser to ten photodetectors. Ten of these optical power splitters are used to form a 10 inlet by 10 outlet optoelectronic switch core. The performance of a typical one to ten optical power splitter is given in table 4.3 [Lam94] disregarding connector losses.

P_{in} (mW)	P_{out} (mW)	Insertion Loss (dB)
4.64	0.36	11.1
	0.37	11.0
	0.40	10.6
	0.41	10.5
	0.40	10.6
	0.42	10.4
	0.41	10.5
	0.37	11.0
	0.38	10.9
	0.45	10.1

Table 4.3 - 1 to 10 Optical Power Splitter Performance

These optical power splitters are passive fiber optic devices and their bandwidth, which is much greater than the electronics used, is not a concern.

To distribute the optical signal to the photodetector arrays, one output from each of the ten splitters is connected each of to the photodetectors in a single array. This forms a ten by ten crossbar switch, allowing any inlet to be connected to any output by turning on the appropriate photodetector. The ten fibers (one output from each of the ten splitters) are epoxied into a silicon holder etched with V-grooves to match the photodetector array spacing. This silicon holder is then aligned and attached to each photodetector array.

Future implementations of optoelectronic switches should use an optical distribution scheme that is more suited for automated manufacture, such as dispensed polymer waveguides [Key94] or lenses [Key95]. These schemes

have the added advantage of being physically much smaller and more robust.

Fiber Delay Lines

The memory requirements to buffer up to N-1 of N incoming packets that are destined for a single output are decidedly non-trivial at gigabit per second data rates. Since any sort of digital logic is very expensive at these bit rates, a large electronic buffer is not economically practical. Lengths of fiber optic strands can be used as the delay lines forming a FIFO (first-in first-out) buffer. An ATM cell at OC-24 data rates requires a fiber (refractive index of 1.45) length of approximately 70.5 meters to delay one cell interval, so the lengths of fiber required are practical.

The stability of the delay provided by the fiber delay lines under changing environmental conditions (especially temperature) is definitely a concern. Other researchers are using fibers for delay lines in spite of this effect [Coh79][Sar90]. The temperature dependence of the delay of an optical pulse can be described by :

$$\frac{dt}{dT} = \frac{1}{c}\left(n\frac{dL}{dT} + L\frac{dn}{dT}\right)$$

$$where \quad \frac{dn}{dT} \approx 1.1 \bullet 10^{-5} \; per \; ^{\circ}C \qquad \text{Eq. 4.2}$$

$$\frac{dL}{dT} \approx 0 \quad in \; comparison$$

where t is the fiber delay, T is the temperature, c is the speed of light, n is the refractive index, and L is the fiber length. From [Coh79], the fiber length change with temperature is negligible in comparison to the refractive index change with temperature over a -40 to 70 degree Celsius temperature range. This leads to a delay change of about 0.4 % of a bit time (2.6 psec) in a 70.5 meter fiber length (one packet delay) at OC-24 data rates (1.244 Gbps) per

degree Centigrade that the temperature changes. This gives a delay change of 4 % of a bit time over a 10 degree Centigrade temperature change for a one cell buffer at any data rate (8% for two cell delay, etc.). In comparison, a 0.5 cm difference in the length of the fiber delay line results in a time difference of 0.024 psec (0.03% of a bit time). For the simple reflex architecture only single packet buffers are allowed, so this delay variation is low enough to be ignored for a laboratory demonstration.

For switches with longer delay lines (i.e. the staggering switch) or that operate in a less controlled environment, this variation can be compensated for by adding a small synchronization FIFO buffer and simply including the delay variation within the packet clock synchronization loop. Thus, as the buffer delay changes, the bits are clocked out of the small synchronization buffer at a slightly different rate so the start of the packets arrive at the photodetector switching elements at the same instant. The synchronization FIFO size required is dependent on the worst case delay variance, since all arriving packets are synchronized at the inputs to the switch, and should be a few bits at most.

Using fiber delay lines to synchronize the packets arriving at the switch on the different inputs requires a variable delay from zero to one packet duration at least. Assuming a temperature controlled fiber length of one cell duration (70.5 meters at OC-24 bit rates), the temperature swing must change by 131,818 degrees Centigrade. A temperature swing of this amount is not practical. Decreasing the required temperature swing to a more realistic 100 degrees Centigrade requires a temperature controlled fiber length of 1318 cell durations (93 kms at OC-24 bit rates). To keep the arriving packets synchronized to within one-tenth of a bit period requires the temperature to be controlled to within 0.023 degrees Centigrade over the entire temperature range. Controlling the temperature this accurately for fibers of these lengths is

not practical.

Photodetector Arrays

An ideal photodetector for use as a switch would have infinite bandwidth when on, block everything when off, add zero noise when on or off, and have an instantaneous turn-on and turn-off time. An ideal array of photodetectors would also have infinite isolation and crosstalk among the photodetectors in the array, as well as being simple to integrate.

Metal-semiconductor-metal (MSM) photodetectors were used in this work because the simple structure of the MSM photodetector is easy to integrate into an array of detectors that form a column of the crossbar switch core [Soo92]. As well, MSM photodetectors have lower capacitance than PIN photodetectors resulting in better performance if contacts transparent at the wavelength of interest is used [Liu95a]. Indium tin oxide (ITO) is a metal that is transparent at the 800 nm wavelength region, and characteristics of MSMs fabricated with ITO will be presented.

The structure of a MSM is a set of interlaced metal fingers deposited upon a semiconductor substrate, with half the fingers connected to a bias voltage, the other half connected to output as shown in figure 4.21. The two sets of metal fingers form a pair of back-to-back Schottky diodes, so that applied voltage of either polarity generates an electric field between the fingers. Electron-hole pairs generated by the incident optical signal are then swept by the electric field to the appropriate finger where they are collected and amplified to generate the electrical signal output. Full width half maximum impulse response times less than 1.2 psec have been measured for MSM photodetectors [Che91a].

a) structure b) schematic

Figure 4.21 - MSM Photodetector - a) Structure, b) Schematic

When a bias voltage (either positive or negative, usually about 5 volts for ease of integration with digital logic) is applied to a MSM photodetector, a current is generated proportional to the optical power incident upon the photodetector. When no bias voltage is applied ideally, no current is generated even when optical power is incident on the photodetector (ideally). In practice, there are effects such as dark current (current generated when no optical power is present), isolation (current generated even when the photodetector is nominally off), optical crosstalk (light aimed at photodetector A overlaps and shines on photodetector B as well), and noise (shot noise, thermal noise, etc.) to worry about as well.

Since MSM photodetectors are symmetric devices, the bias voltage can be either positive or negative unlike a PIN photodetector. Due to small fabrication differences between the Schottky contacts on each of the fingers, the bias voltage giving the highest isolation may be slightly different from the ideal of zero volts. This was demonstrated in [Lam94] where the highest isolation of 50 dB for a MSM photodetector was achieved with a bias voltage of +12.7 mV.

The switch core requires an array of photodetectors that have individual bias controls and have their outputs summed into a single amplifier as shown in figure 4.22. When the bias is controlled directly, as shown in the figure, the individual capacitances of each of the MSM photodetectors are added, forming

a larger capacitance that will decrease the maximum achievable bandwidth of the array of photodetectors. By changing the bias arrangement to that shown in figure 4.23, only the capacitance of the single active photodetector is important. This allows much larger arrays of photodetectors that still have large bandwidths [RIM94]. As well, the inactive photodetectors don't contribute to the noise of the system and reduce the overall sensitivity [For89] as they would if the inactive photodetectors had their bias grounded.

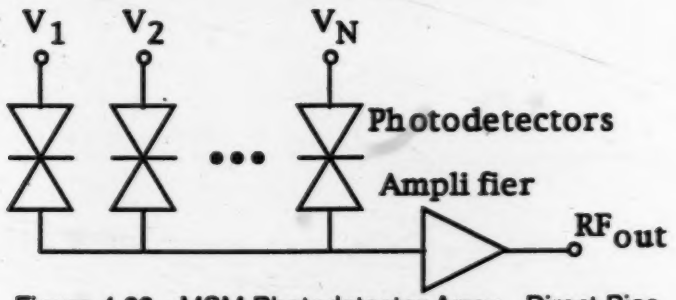


Figure 4.22 - MSM Photodetector Array - Direct Bias

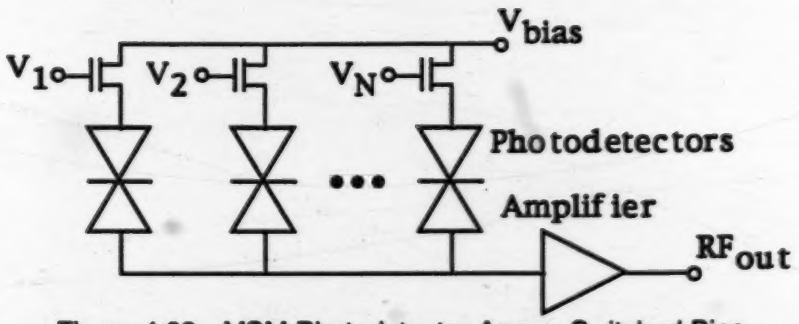


Figure 4.23 - MSM Photodetector Array - Switched Bias

Bandwidth measurements on several directly biased MSM photodetector arrays were performed using the HP 8753A/HP 85046A test equipment with a 3 GHz upper bandwidth limit in the test setup shown in figure 4.24. A representative example is shown in figure 4.25 which demonstrates that the laser is the bandwidth limiting device in the system. All the directly biased MSM photodetector arrays performed quite similarly in terms of bandwidth. The MSM photodetector array fabricated with ITO metal fingers, which are transparent to

800 nm wavelength light, showed a higher responsivity as seen in figure 4.26, although the bandwidth remains very similar.

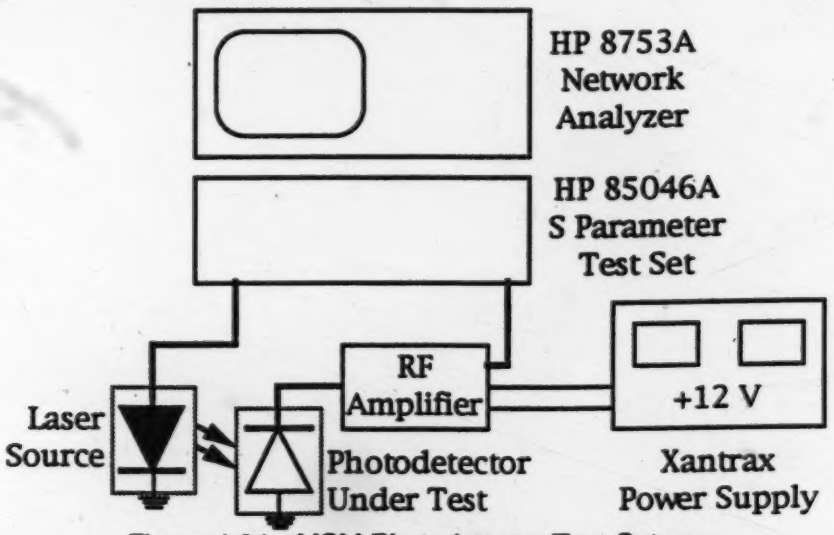


Figure 4.24 - MSM Photodetector Test Setup

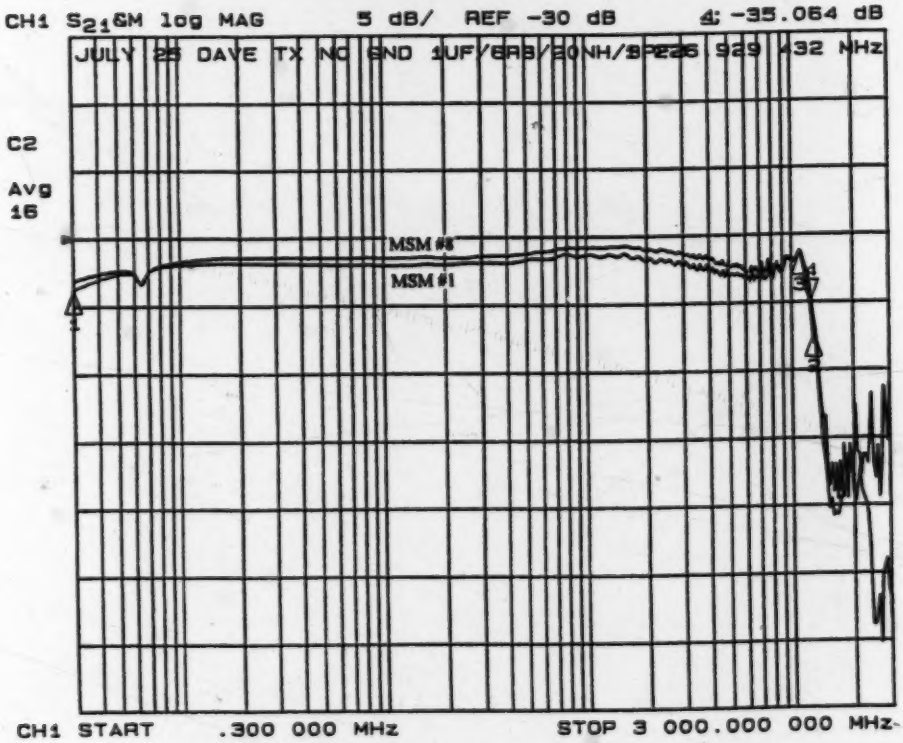


Figure 4.25 - MSM Photodetector Array Bandwidth

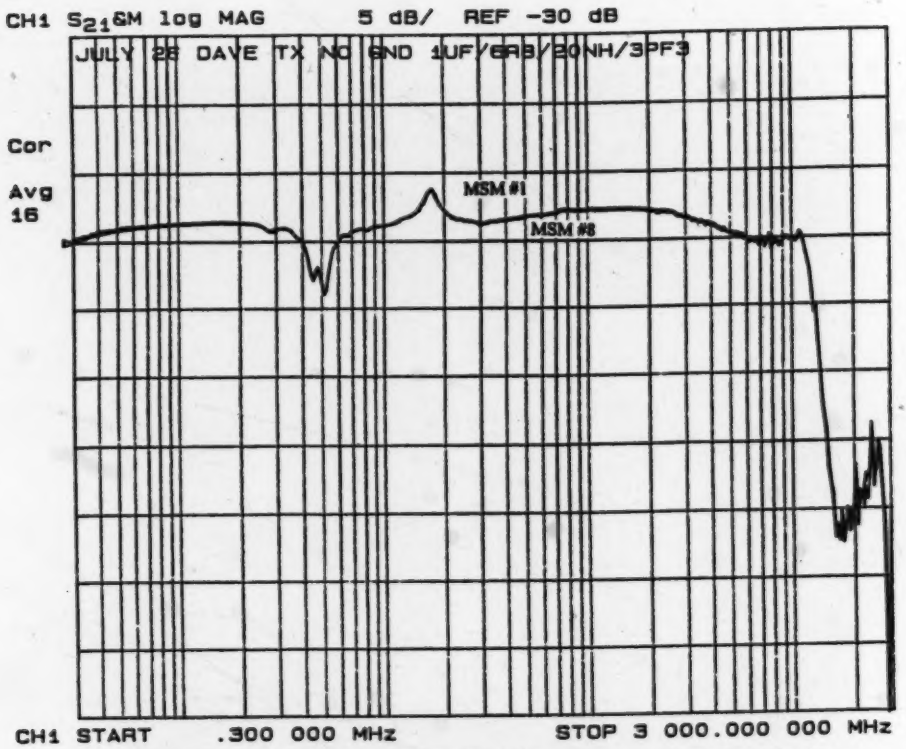


Figure 4.26 - ITO MSM Photodetector Array Bandwidth

A closer look was taken at the bandwidth response of the directly biased MSM photodetector arrays using a HP 8510B network analyzer (26 GHz bandwidth), United Technologies Photonics Mach-Zehnder Modulator (18 GHz bandwidth), and a Mitec 18 GHz wide band amplifier. This work was undertaken by Rohit Sharma, Ray DeCorby and myself to determine the achievable bandwidth of the system if a higher bandwidth laser source were to be developed. The measurements show a remarkable change in bandwidth dependent on the position of the MSM photodetector being measured within the array as detailed in [TRL94d]. A sample of the change is shown in figures 4.27 and 4.28, showing the bandwidth measured for an MSM photodetector close to the 50 Ω output and an MSM photodetector farther from the 50 Ω output. Simulations performed by Rohit Sharma [TRL94e] and Qing Liu [Liu95b] isolated the cause of the problem as the capacitance of the MSM

photodetectors in combination with the inductance of the connecting strip lines in the MSM photodetector array. Changing the layout of the MSM photodetector array appropriately will minimize these effects. [Liu95b] has proposed one possible solution.

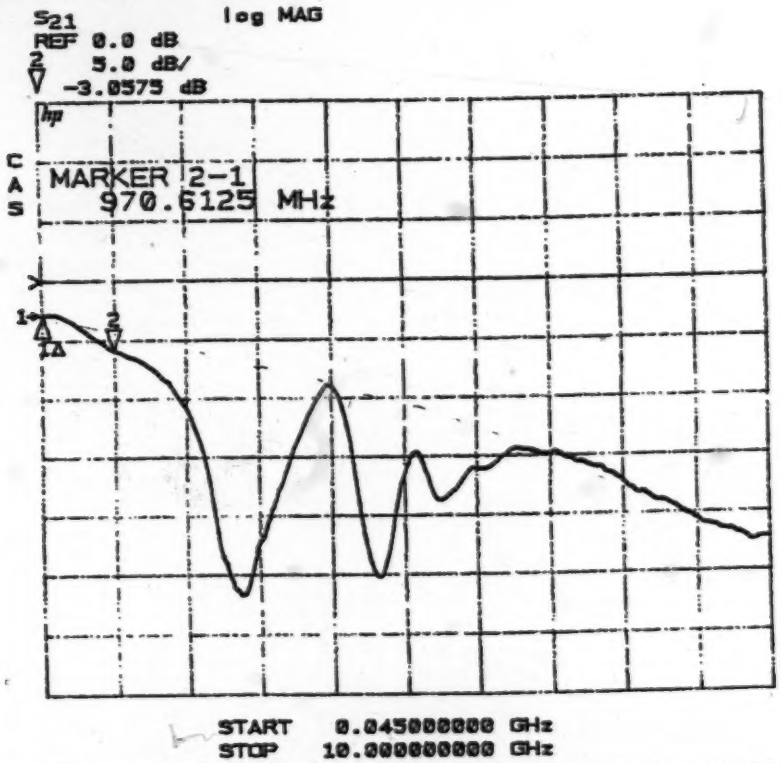


Figure 4.27 - MSM #1 of 8 Photodetector Array Bandwidth

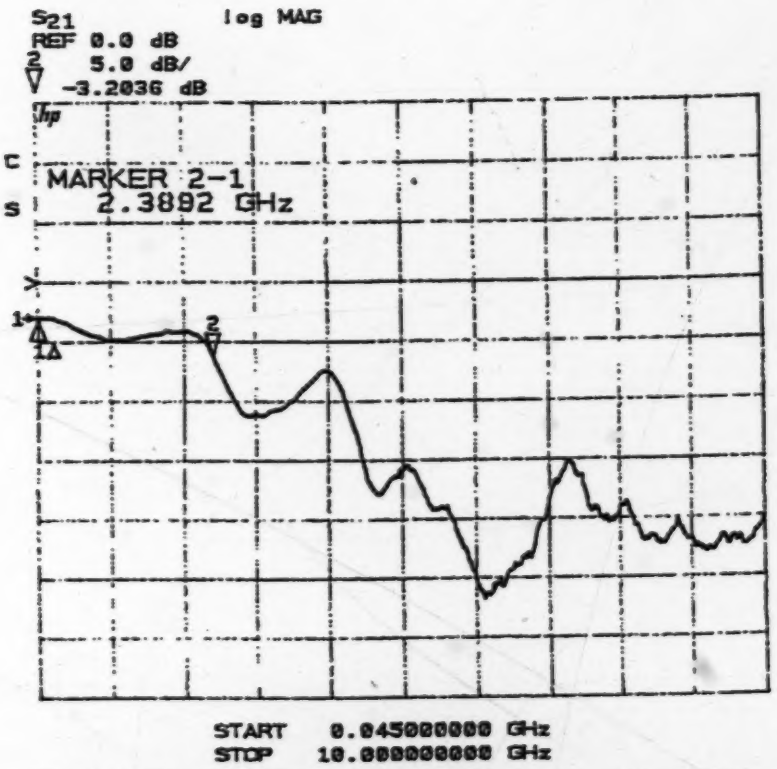


Figure 4.28 - MSM #7 of 8 Photodetector Array Bandwidth

Photodetector isolation refers to the ratio of the power in the response when a single photodetector is enabled and when it is disabled. The same test setup was used to measure the isolation of a photodetector as to measure the bandwidth, with the difference of all photodetectors in the array being illuminated. Typical results are shown in figure 4.29 from [Lam94] giving an isolation of 50 dB or more. The four lines, from highest response to lowest response, are +5 V bias, 0 V bias (grounded), +12.7 mV bias, and an open circuit. A similar result was obtained using a MSM photodetector array similar to those described in [Gou94] as shown in figure 4.30. This array has an embedded GaAs transimpedance amplifier attached to the common output from the MSM photodetector array. Due to the reactive ion etching fabrication technique used for these devices which caused a large amount of surface damage, the MSM photodetector bandwidth was limited to less than 0.5 GHz

[Vei92]. The four lines in this figure show the measured response when the photodetector under test was biased at +5 V, -5 V, 0 V, and open circuit. The embedded amplifier was designed to use a +5 V bias on the MSM photodetectors in the array, leading to the poor result for the -5 V bias. As expected, the open circuit produces the best isolation of greater than 40 dB.

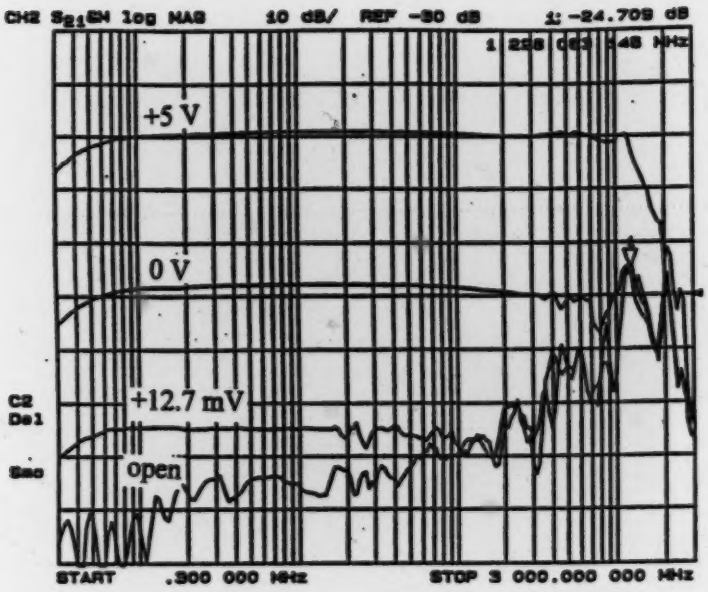


Figure 4.29 - MSM Photodetector Array Isolation ([Lam94])

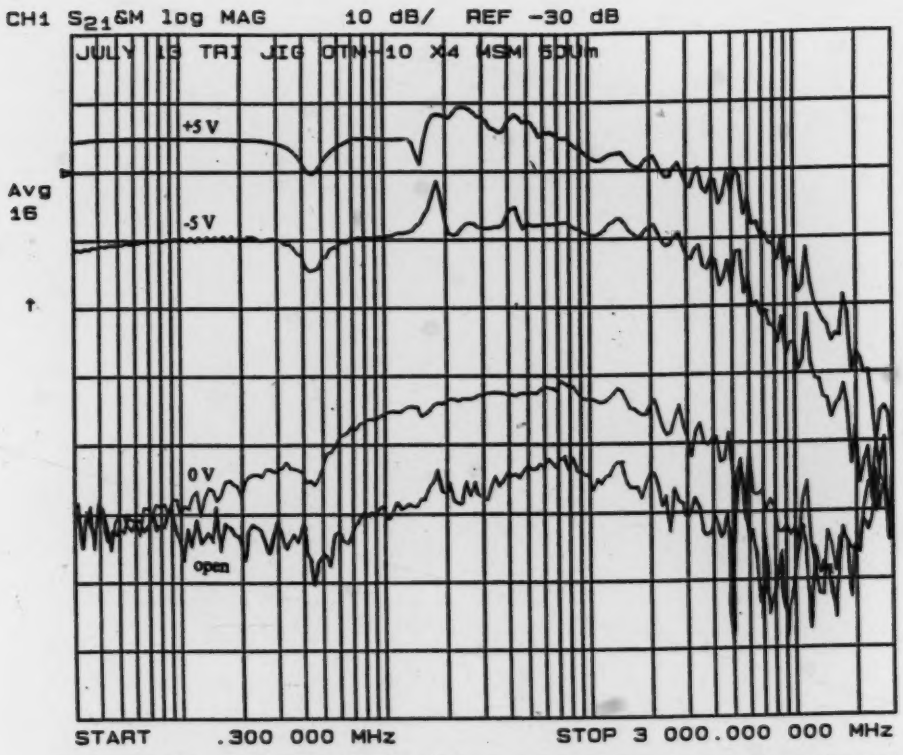
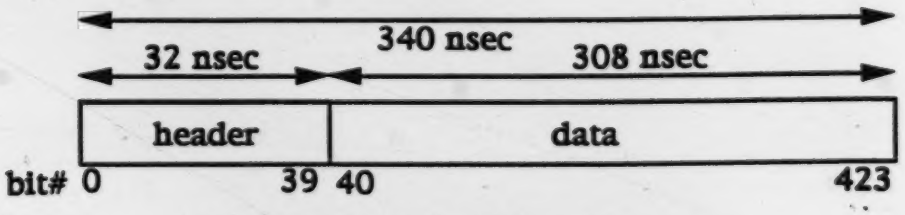


Figure 4.30 - MSM Photodetector Array Isolation (similar to [Gou94])

Initially, the MSM photodetector switching speed was thought to be an important factor in determining the maximum packet rate through the switch. To keep an acceptable bit error rate, the MSM photodetector should switch states completely within ten percent of a bit time. At OC-24 rates, a single bit time is 803 psec, giving a required switching time of 80 psec. This is a very stringent requirement that, fortunately, is not required.



OC-24 rate 1 bit time = 803 psec

Figure 4.31 - ATM Cell Switching Time

For ATM traffic, the cell header is replaced at every output of the switch to

use the new VCI/VPI values. This allows the entire header to be used as the switching time, since the cell header must be regenerated at the output of the switch to put in the new VCI/VPI values. As shown in figure 4.31, this increases the switching time to 32 nsec. The same technique of switching during unused data can also be used for other fast packet switching standards that use a fixed bit pattern at the start of end of a packet for framing.

One of the earliest experiments on optoelectronic switching used an array of PIN photodetectors, with a switching speed of 100 nsec [RIM78]. Other experiments, using GaAs photoconductors for easier integration and current summing, obtained switching speeds of less than one nsec [Lam84][RIM89]. A later experiment built on this work to demonstrate an eight by eight MSM photodetector based optoelectronic crossbar switch that had a reconfiguration time of 100 nsec [For89]. The 100 nsec limit on the switching speed was due to the stray capacitance introduced by the hybrid construction techniques used and was not a limitation of the photodetectors. A later experiment [Ton94] showed a 10% - 90% switching time for an MSM photodetector to be 330 psec, although the ringing induced stretched this to 1 to 2 nsec total dead time.

Attempts were made to determine the minimum switching time for MSM photodetectors. The design of the MSM photodetector array mounting platform has the bias voltages delivered through filtering components (capacitors and ferrite beads) to the individual MSM photodetectors to reduce the noise for analog filtering applications. This design is not suitable for an optoelectronic crossbar switch which must switch the bias to the individual photodetectors quickly. For this reason, a different mounting platform was used which had another 50 Ω RF line in addition to the RF output. This additional line was used to control one of the FETs embedded within the MSM photodetector array. Unfortunately, the embedded FETs were not functional on the three arrays that

were wire-bonded into this mounting platform and no results were obtained for the switching time of MSM photodetectors.

Address Replacement

The address replacement block is used to replace the existing VCI/VPI in the cell header with the new VCI/VPI. The new CRC for the header must be calculated within the time occupied by this block. This is a peripheral block in the switch architecture that has the same requirements independent of the technology used within the switch core and will not be discussed in detail. The usual solution is to have the new VCI/VPI retrieved from the associative memory that determines the desired output. These values are then tracked by the controller logic and inserted into the cell header at the desired output. Since this block only operates at the packet rate (not the bit rate) of an input, CMOS logic and memories [Qui94] can meet the performance requirements.

For the case where the switching element corrupts data in the header, the additional bits in the cell header must also be preserved for insertion at the desired output. This is a relatively simple addition of four bits (payload type and cell loss priority bits) that the controller must preserve in addition to the 28 bits for the output VCI/VPI.

Simple Output Queues

The simple output queue block forms a simple FIFO buffer that operates at the line rate to buffer cells at each output. The fiber delay lines form a shared output queue that is used for buffering when the small simple output queue overflows at one or more of the outputs. Unlike a shared output queue the simple output queue only needs to accept one arriving packet per switch core time slot. Inexpensive CMOS logic will be adequate. Again, this is a peripheral block that is used in any switch architecture that requires simple output queues and will not be examined further.

Controller Design

The controller operates at the packet rate, and must determine the settings for all the crosspoints in the switch core within a single packet time. At the OC-24 rate, an ATM cell has a duration of just over 340 nsec within the switch core, so the controller required by the simple reflex switch can be implemented in inexpensive CMOS logic. For these rates, a software based controller (such as a digital signal processor) will not be economically feasible.

The simplest controller mechanism is a large state machine implemented in a read only memory (ROM). This controller type is feasible only for smaller switch sizes. Given a switch with N inputs, M outputs, P fiber delay line buffers, N+P inlets, and M+P outlets the resulting controller ROM size is given by equation 4.3. The two terms making up the number of inputs to the ROM controller are the (N+P) term showing one set of desired output states for each inlet, and the $\log_2(M+1)$ term indicating the number of states to indicate the desired output plus one for an idle cell at the inlet. Two terms also make up the number of outputs from the ROM controller, with the (M+P) term showing one inlet to connect for each of the outlets, and the $\log_2(N+P+1)$ term showing the actual inlet connected for this outlet. For a small switch with 4 inputs, 4 outputs, and 4 fiber delay lines works out to a ROM controller size of 2^{24} x 32 bits. A larger switch with 8 inputs, 8 outputs, and 8 fiber delay lines, the ROM controller size is an unfeasible 2^{64} x 80 bits. Note that this ROM controller will return all the crosspoint settings in a single cycle, so the ROM access time has to be the packet time or less.

$$2^{(N+P)\lceil \log_2(M+1) \rceil} \times (M+P)\lceil \log_2(N+P+1) \rceil \qquad \text{Eq. 4.3}$$

To reduce the ROM required, the controller can be designed to require a single access for each outlet, for a total of M+P accesses to set all the

crosspoints. Equation 4.4 shows the ROM controller size required, with the new M+P term holding the busy or idle status of the outlets. For the 4 input square switch with 4 buffers the ROM controller size is 2^{11} x 4 bits, while the 8 input square switch with 8 buffers requires a ROM controller of 2^{20} x 5 bits. The drawback is the smaller switch requires 8 accesses in a single packet time, while the larger switch requires 16 accesses in a single packet time.

$$2^{M+P+\lceil \log_2(M+1) \rceil} \times \lceil \log_2(N+P+1) \rceil \qquad \text{Eq. 4.4}$$

The obvious next step is to develop a ROM controller that sets more than one crosspoint state in a single access but not all at once. Assuming the controller takes b cycles to set all the crosspoints, it will set a inlets per cycle where a is given by equation 4.5. Equation 4.6 shows the ROM controller size required. For the sample 8 input switch with 8 buffers and assuming 4 cycles, a ROM of 2^{32} x 16 bits is required. If 6 cycles are assumed, the required ROM size is reduced to 2^{28} x 15 bits.

$$a = \left\lceil \frac{N+P}{b} \right\rceil \qquad \text{Eq. 4.5}$$

$$2^{M+P+a\lceil \log_2(M+1) \rceil} \times a \lceil \log_2(N+P+1) \rceil \qquad \text{Eq. 4.6}$$

The above ROM controller sizes assume single-cast packets, meaning the packets arriving at the inputs are destined for only a single output. Multicast packets, packets destined for more than one output, are useful for implementing broadcast packet distribution schemes. Multicast packets are easily accommodated by the ROM controller, at the expense of increasing the ROM size required by transforming the $\log_2(M+1)$ term to a simple M term. A multicast packet that is required to be stored in a fiber delay line buffer for certain outputs can be stored in a single delay line and transferred to the

desired output(s) when they are available. Other switch architectures deal with multicast packets by replicating the packet at the input and sending in extra single-cast packets to simulate the multicast packet. This is undesirable because it increases the traffic within the switch core.

Conclusions

In this thesis, three subjects were discussed. First, the architectures of fast packet switches were examined in terms of their components and switch core technology. Second, by means of simulations, various switch architectures were compared based on cell loss probability and latency. Finally, experimental measurements were performed on key components that make up the 'best' switch architecture as decided in the second section.

Switch Architectures

Switch architectures were first compared based on the location of the required buffering. The various options were discussed, with shared output buffering being the optimal choice in terms of buffer size and delay characteristics. Switch architectures were then compared based on the method used to connect the inputs to the outputs. The choice was made to concentrate on space division switching with passive signal distribution due to its ability to support higher bit rates and the availability of relatively inexpensive, reliable components.

The next comparison focused on the technology used to connect the inputs to the outputs. All-electrical switching was not considered because of its bandwidth limitations. All-optical switching was dismissed because the components are expensive, not easily available, and are not easily integratable at the current time. Optoelectronic switching was then considered, either transmitter or receiver switched. Receiver-switched optoelectronic matrices have several advantages over the other technologies considered, including inexpensive components that are easily integrated into arrays used to form a switch core.

Switch Simulations

Various switch architectures were simulated to determine their packet loss probability and latency characteristics. Two simple switch architectures, the crossbar and knockout switches, were simulated first. Theoretical results that were in extremely good agreement with the simulated results were calculated for these switches lending confidence to the simulation program used. A few other switch architectures from the literature were then simulated to ensure the simulation results agreed with the published results.

A fast packet switch architecture was then developed to take advantage of the unique strengths of receiver-switched optoelectronic switching and fiber delay lines. These strengths include the fact that the output is electronic, the controller is simple and each individual crosspoint is inexpensive. It was also demonstrated that this architecture could be improved by operating the switch core at a faster rate than the inputs. These switch architectures were then compared to one another in terms of the packet loss performance and latency characteristics.

Switch Components

The components that make up the fast packet switch were discussed next. Common components, those that are required for any fast packet switch whether optical, electronic, or optoelectronic, were only briefly discussed. Laboratory measurements were performed on the electrical amplifiers, lasers, and MSM photodetectors to be used in the fast packet switch. The amplifiers, lasers, and MSM photodetector arrays were shown to be capable of 1.2 GHz bandwidth, easily allowing OC-24 rate operation. Unfortunately, the switching time measurements were not successful so a measured value for the MSM photodetector on/off time was not verified, although results of 2 nsec are in the literature [Ton94] which are limited by stray capacitance not the MSM

photodetectors themselves.

A possible controller design was discussed next. This controller design is simple, easy to construct, and allows single-cast and multicast packets without forcing packet replication.

Future Work

From this project, the following areas require further investigation :

A more accurate data traffic model is required to compare the various switch architectures under realistic conditions for data traffic.

Experimental work to characterize the noise performance of the optoelectronic data path as well as the switching time of the MSM photodetectors is required to determine the amount of cycling possible in an optoelectronic reflex switch.

A special controller will be required to allow the full scale demonstration of a fast packet switch using the existing analog 10 by 10 optoelectronic switch matrix.

The effect of multistage switches needs to be carefully investigated experimentally to extend the work in [Lam94] and [RIM94] from the analog to the digital domain.

Bibliography

Aca87 Acampora, A, "A Multichannel Multihop Local Lightwave Network", Globecom '87, 1987.

Alb90 Albertengo, G and Sisto, R, "Parallel CRC Generation", IEEE Micro, Oct 1990, pgs 63-71.

Art88 Arthurs, E, Goodman, MS, Kobrinski, H and Vecchi, MP, "HYPASS: An Optoelectronic Hybrid Packet Switching System", IEEE Journal on Selected Areas in Communications, Vol 6, No 9, Dec 1988, pgs 1500-1510.

Bal94 Ball, GA, Glenn, WH, and Morey, WM, "Programmable Fiber Optic Delay Lines", IEEE Photonics Technology Letters, Vol 6, No 6, June 1994, pgs 741-743.

Bit91 Bitmead, RR, Johnson, CR, and Pollock, CR, "Optical Adaptive Signal Processing: An Appraisal", Int. Journal of Adaptive Control and Signal Processing, Vol 5, 1991, pgs 87-92.

Bor93 Borgonovo, F, Fratta, L and Bannister, J, "Unslotted Deflection Routing in All-Optical Networks", Globecom '93, Houston, TX, Nov 29-Dec 2, 1993, pgs 119-125.

Bri94 Briggman, D, Hanke, G, Langmann, U and Pottbacker, A, "Clock Recovery Circuits up to 20 Gbit/s for Optical Transmission Systems", IEEE MTT-S Digest, paper WE3F-44, San Diego, CA, May 23-27, 1994, pgs 1093-1096.

Bur92 Burke, C, Fujiwara, M, Yagamuchi, M, Nishimoto, H and Honmou, H, "128 Line Photonic Switching System Using LiNbO3 Switch Matrices and Semiconductor Traveling Wave Amplifiers", Journal of Lightwave Technology, Vol 10, No 5, May 1992, pgs 610-615.

Che91a Chen, Y, Brock, T, Smith, FW and Calawa, AR, "375 GHz Bandwidth Photoconductive Detector", Applied Physics Letters, Vol 59, No 16, Oct 14, 1991, pgs 1984-1986.

Che91b Chen, DX and Mark, JW, "SCOQ: A Fast Packet Switch with Shared Concentration and Output Queueing", Infocom '91, Bal Harbour, FL, Apr 7-11, 1991, pgs 145-154.

Cho92 Chou, SY and Liu, MY, "Nanoscale Tera-Hertz Metal-Semiconductor-Metal Photodetectors", IEEE Journal of Quantum Electronics, Vol 28, No 10, Oct 1992, pgs 2358-2368.

Cis91 Cisneros, A and Brackett, CA, "A Large ATM Switch Based on Memory Switches and Optical Star Couplers",. IEEE Journal on Selected Areas in Communications, Vol 9, No 8, Oct 1991, pgs 1348-1359.

Coh79 Cohen, LG and Fleming, JW, "Effect of Temperature on Transmission in Lightguides", Bell System Technical Journal, Vol 58, No 4, Apr 1979, pgs 945-950.

Com96 Comerford, Richard, ed. "It's only growing pains - The second annual roundtable on the state of the Internet", IEEE Spectrum, Sept. 1996, pgs 46-55.

Cou87 Coudreuse, J-P and Servel, M, "Prelude: An Asynchronous Time Division Switched Network", ICC '87, Seattle, Wash., June 1987, pgs 769-783.

Cra95 Craig, Ian and McFarlane, John, "Fast Forwading Into The Future: Broadband's Bright Promise Beckons", Telesis, Issue 100, Oct, 1995, pgs 119-120.

DeZ94 De Zhong, W, Tsukada, M, Yukimatsu, K and Shimazu, Y, "Terahipas: A Modular and Expandable Terabit/second Hierarchically Multiplexing Photonic ATM Switch Architecture", Journal of Lightwave Technology, Vol 12, No 7, July 1994, pgs 1307-1315.

Dod93 Dodds, DE and Du, L, "ATM Framing Acquisition", WESCANEX '93, Saskatoon, SK, Canada, May 17-18, 1993, pgs 56-60.

Don90 Dono, NR, Green Jr, PE, Ramaswami, R and Tong F, "A Wavelength Division Multiple Access Network for Computer Communications", IEEE Journal on Selected Areas in Communications, Vol 8, 1990, pgs 983-994.

dPr87 dePrycker, M and Bauwens, J, "A Switching Exchange for an Asynchronous Time Division based Network", ICC '87, Seattle, June 1987.

dPr91 dePrycker, Martin, "Asynchronous Transfer Mode: Solution for Broadband ISDN", Ellis Horwood, 1991, ISBN 0-13-053513-3.

dPr95 dePrycker, Martin, "Asynchronous Transfer Mode: Solution for Broadband ISDN Third Edition", Ellis Horwood, 1995, ISBN 0-13-342171-6.

Eck88 Eckberg, AE and Hou T-C, "Effects of Output Buffer Sharing on Buffer Requirements in an ATM Packet Switch", Infocom '88, paper 5A.4, 1988, pgs 459-466.

Ell93 Ellis, AD, Ellis, K and Patrick, DM, "All Optical Clock Recovery at Bit Rates up to 40 Gbit/s", Electronics Letters, Vol 29, No 15, July 22, 1993, pgs 1323-1324.

Ena92 Enam, SK and Abidi, AA, "MOS Decision and Clock Recovery Circuits for Gbps Optical Receivers", ISSCC '92, San Francisco, CA, Feb 1992.

For89 Forrest, SR, Tangonan, GL and Jones, V, "A Simple 8x8 Optoelectronic Crossbar Switch", Journal of Lightwave Technology, Vol 7, No 4, Apr 1989, pgs 607-614.

FOR93 "Forerunner ASX-100 Specifications", Fore Systems Inc., June 1993.

Fre78 Freedman, D, Pisani, R and Purves, R, "Statistics", W.W. Norton and Company, New York, 1978, ISBN 0-393-09076-0.

Fre89 Freeman, Roger L., "Telecommunication System Engineering", Second Edition, John Wiley & Sons, 1989, ISBN 0-471-63423-9.

Gia91 Giacopelli, JN, Hickey, JJ, Marcus, WS, Sincoskie, WD, and Littlewood, M, "Sunshine: A High-Performance Self-Routing Broadband Packet Switch Architecture", IEEE Journal on Selected Areas in Communications, Vol 9, No 8, Oct 1991, pgs 1289-1298.

Gou94 Gouin, F, Bealieu, C and Noad, J, "Design, Fabrication and Characterization of High Speed GaAs MSMs for OEIC Applications", International Conference on Applications of Photonic Technology, Toronto, Ont, Canada, June 21-23, 1994.

Gra94 Granestrand, P, Lagerstrom, B and Svensson, P..., "Pigtailed Tree Structured 8x8 LiNbO3 Switch Matrix with 112 Digital Optical Switches", IEEE Photonic Technology Letters, Vol 6, No 1, Jan 1994, pgs 71-73.

Gus93 Gustavsson, M, Janson, M and Lundgren, L, "Digital Transmission Experiment with Monolithic 4x4 InGaAsP/InP Laser

Amplifier Gate Switch Array", Electronics Letters, Vol 29, No 12, June 10, 1993, pgs 1083-1085.

Haa92a Haas, Z, "Optical Slot Synchronisation Scheme", Electronics Letters, Vol 28, No 23, Nov 5, 1992, pgs 2184-2185.

Haa92b Haas, Z, "The 'Staggering Switch': An 'Almost-All' Optical Packet Switch", OFC '92, San Jose, CA, Feb 2-7, 1992.

Hlu88 Hluchyj, MG and Karol, MJ, "Queueing in High-Performance Packet Switching", IEEE Journal on Selected Areas in Communications, Vol 6, No 9, Dec 1988, pgs 1587-1597.

HP75 HP 150-7, "Spectrum Analysis ... Signal Enhancement", Hewlett Packard, June 1975.

Hua84 Huang, A and Knauer, S, "Starlite: A Wideband Digital Switch", Globecom '84, Atlanta, Georgia, Nov 26-29, 1984, pgs 401-409.

Hui87 Hui, JY and Arthurs, E, "A Broadband Packet Switch for Integrated Transport", IEEE Journal on Selected Areas in Communications, Vol 9, No 8, Oct 1987, pgs 1264-1273.

Hur91 Hurm V et al, "10 Gbit/s Monolithic Integrated Optoelectronic Receiver using an MSM Photodiode and AlGaAs/GaAs HEMTs", Microelectronic Engineering, Vol 15, 1991, pgs 275-278.

ITU93 ITU-T Recommendation I.361, "B-ISDN ATM Layer Specification", International Telecommunications Union, Mar 1993.

Jai94 Jain, Raj K., "ATM Networks: Issues and Challenges Ahead", ATM: Technology, Standards, Trials, and Applications, Vancouver, BC, Canada, Jan. 27-28, 1994.

Jan94 Janz, CF, Keyworth, BP, Allegretto, W, MacDonald, RI, Fallahi, M, Hillier, G and Rolland, C, "Mach-Zehnder Switch using an Ultra-Compact Directional Coupler in a Strongly-Confining Rib Structure", Photonics Technology Letters, Vol 6, No 8, Aug 1994, pgs 981-983.

Jin92 Jinno, M, "Ultrafast Time-Division Demultiplexer Based on Electro-optic On/Off Gates", Journal of Lightwave Technology, Vol 10, No 10, Oct 1992, pgs 1458-1465.

Kar87 Karol, MJ, Hluchyj, MG and Morgan, SP, "Input Versus Output Queueing on a Space Division Packet Switch", IEEE Transactions on Communications, Vol 35, No 12, Dec 1987, pgs 1347-1356.

Kar93 Karol, MJ, "Shared-Memory Optical Packet (ATM) Switch", SPIE Vol 2024 Multigigabit Communication Systems, 1993, pgs 212-222.

Key94 Keyworth, BP, Corazza, D, Fuchs, W and McMullin, JN, "Low-Cost Direct-Write Multimode Polymer Waveguides", LEOS '94 Summer Topical Meeting, Lake Tahoe, NV, July 6-8, 1994.

Key95 Keyworth, BP, McMullin, JN, Corazza, D, Mobbott, L, Neufeld, T and Rosadiuk, T, "Dispensed Polymer Microlenses", Technical Digest Optical Society of America Annual Meeting, MJJ5, Sept 10-15, 1995, pg 61.

Koh94 Kohama, Y, Ohiso, Y and Kurokawa, T, "0.85 um bottom-emitting vertical-cavity surface-emitting laser diode arrays grown on AlGaAs substrates", Electronics Letters, Vol 30, No 17, Aug 18, 1994, pgs 1406-1407.

Lam84 Lam, DKW and MacDonald, RI, "A Reflex Optoelectronic Switching Matrix", Journal of Lightwave Technology, Vol 2, No 2, Apr 1984, pgs 88-90.

Lam94 Lam, D, "Optoelectronic Reflex Signal Processor", M.Sc. Thesis, University of Alberta, Edmonton, AB, Canada, 1994.

Lel94 Leland, WE, Taqqu, MS, Willinger, W and Wilson, DV, "On the Self-Similar Nature of Ethernet Traffic (Extended Version)", IEEE/ACM Transactions on Networking, Vol 2, No 1, Feb 1994, pgs 1-15.

Liu95a Liu, QZ, "Sensitivity Comparison of MSM-HBT and p-i-n-HBT OEIC Receivers", Microwave and Optical Technology Letters, Vol 9, No 5, Aug 5, 1995, pgs 263-266.

Liu95b Liu, QZ, "Bandwidth Enhancement of a Metal-Semiconductor-Metal Photodetector Array", Microwave and Optical Technology Letters, Vol 10, No 4, Nov 1995, pgs 238-241.

Low93 Lowy, M, "A Low Power Architecture for Digital Sequence Generation", Globecom '93, Houston, TX, Nov 29-Dec 2, 1993, pgs 636-640.

McQ91 McQuillan, JN, "Cell Relay Switching", Data Communications, Sept 1991, pgs 58-69.

Mel92 Melen, R, "Current Architectures for ATM Implementation", ETT, Vol 3, No 2, Mar-Apr 1992, pgs 145-155.

Mid92 Midwinter, JE, "Photonics in Switching: the Next 25 Years of Optical Communications", IEE Proceedings, Part J, Vol 139, No 1, Feb 1992, pgs 1-12.

Mid93a Midwinter, JE, "Photonics in Switching, Volume I Background and Components", Academic Press, 1993, ISBN 0-12-496051-0.

Mid93b Midwinter, JE, "Photonics in Switching, Volume II Systems", Academic Press, 1993, ISBN 0-12-496052-9.

New92 Newman, P., "ATM Technology for Corporate Networks", IEEE Communications Magazine, Vol 30, No 4, Apr 1992, pgs 90-101.

Nis93 Nishi, T, Yamamoto, T and Kuroyanagi, S, "A Polarization Controlled Free Space Photonic Switch Based on a PI-LOSS Switch", IEEE Photonics Technology Letters, Vol 5, No 9, Sept 1993, pgs 1104-1106.

Obr94 Obro, M, Thorsen, P and Andreasen, SB, "All-optical frame synchronisation recovery", Electronic Letters, Vol 30, No 15, July 21, 1994, pgs 1243-1244.

Pat93 Pattavina, A and Bruzzi, G, "Analysis of Input and Output Queueing for Nonblocking ATM Switches", IEEE/ACM Transactions on Networking, Vol 1, No 3, June 1993, pgs 314-328.

Pat94 Patrick, DM and Manning, RJ, "20 GBit/s all-optical clock recovery using semiconductor nonlinearity", Electronics Letters, Vol 30, No 2, Jan 20, 1994, pgs 151-152.

Pet88 Petermann, K, "Laser Diode Modulation and Noise", Kluwer Academic Press, 1988, ISBN 90-277-2672-8.

Qui94 Quinnell, RA, "Synchronous Memories", EDN, Aug 4, 1994, pgs 56-66.

RIM78 MacDonald, RI and Hara, EM, "Optoelectronic Broadband Switching Array", Electronics Letters, Vol 14, No 16, Aug 3, 1978, pgs 502-503.

RIM88 MacDonald, RI, "Terminology for Photonic Matrix Switches", IEEE Journal on Selected Areas in Communications, Vol 6, No 7, Aug 1988, pgs 1141-1151.

RIM89 MacDonald, RI, "Optoelectronic Matrix Switching", Canadian Journal of Physics, Vol 67, 1989, pgs 389-393.

RIM94 MacDonald, RI, Lam, D and Noad, J, "Multistage optoelectronic switch networks", IEE Proceedings of Optoelectronics, Vol 141, No 3, June 1994, pgs 173-177.

Sar90 Sarrazin, DB, Jordan, HF and Heuring VP, "Fiber optic delay line memory", Applied Optics, Vol 29, No 5, Feb 10, 1990, pgs 627-637.

Smi92 Smith, K and Lucek, JK, "All-Optical Clock Recovery Using a Mode-Locked Laser", Electronics Letters, Vol 28, No 19, Sept 10, 1992, pgs 1814-1816.

Soo92 Soole, JBD, "InGaAs MSM Photodetectors for Long Wavelength Fiber Communications", SPIE Vol 1680, High-Speed Electronics and Optoelectronics, 1992, pgs 153-160.

Tac94 Tachikura, M, Katagiri, T and Kobayashi, H, "Strictly Nonblocking 512x512 Optical Fiber Matrix Switch Based on Three Stage Clos Network", IEEE Photonics Technology Letters, Vol 6, No 6, June 1994, pgs 764-766.

The88 Thevenaz, L, Pellaux, J-P and Von Der Weid, J-P, "All-Fiber Interferometer for Chromatic Dispersion Measurements", Journal of Lightwave Technology, Vol 6, No 1, Jan 1988, pgs 1-7.

Tob90 Tobagi, FA, "Fast Packet Switch Architectures for Broadband ISDN", Proceedings of the IEEE, Vol 78, No 1, Jan 1990, pgs 133-167.

Ton94 Tong, F, Kwark, YH and Stevens, AE, "A Four-Channel Monolithic Optical/Electronic Selector for Fast Packet-Switched WDMA Networks", IEEE Photonics Technology Letters, Vol 6, No 1, Jan 1994, pgs 68-70.

TRI92 "TQ8016 16x16 Digital Crosspoint Switch", Triquint Semiconductor, Digital Communications and Signal Processing, Oct. 1992.

TRL94a Clegg, D, Sharma R, and Tholl R, "RF Amplifier 01", TRLabs internal specification report, 1994.

TRL94b Clegg, D, Tholl, R, and Sharma R, "RF Amplifier 02", TRLabs internal specification report, 1994.

TRL94c Clegg, D, "Laser Driver and RF Bias", TRLabs internal specification report, 1994.

TRL94d Sharma, R, Tholl, R, DeCorby, R, and Clegg, D, "Frequency Response Characteristics of GaAs Metal-Semiconductor-Metal (MSM) Photodetector Arrays", TRLabs internal report TR-94-03, Apr. 1994.

TRL94e Liu, QZ, "Analysis and Simulations of GaAs Metal Semiconductor Metal Photodetector Array", TRLabs internal report TR-94-04, May 1994.

Tsu94 Tsukada, M, Misawa, A, Nishikido, J, Shimazu, Y and Nakano, H, "Experiments on Photonic Cell Switching with an Optical Input Buffer", Electronics Letters, Vol 30, No 13, June 23, 1994, pgs 1081-1082.

UTP94 "800nm APE Y-fed Balanced Bridge Modulator (High Frequency, Dual Output, Transformer Matched, Intensity Modulator)", UT Photonics, Inc., Feb. 1994.

Vai88 Vaidya, AK and Pashan MA, "Technology Advances in Wideband Packet Switching", Globecom '88, paper 21.3, Hollywood, FL, Nov 28-Dec 1, 1988, pgs 68-671.

Vei90 Veilleux, M, "Construction for a 10x10 Broadband Matrix Switch", M.Sc. Thesis, University of Alberta, 1990.

Vei92 Veilleux, M, private communication.

Wei94 Weich, K, Patzak, E and Horer J, "Fast all-optical switching using two-section injection -locked semiconductor lasers", Electronics Letters, Vol 30, No 6, JMar 17, 1994, pgs 493-494.

Yeh87 Yeh, Y-S, Hluchyj, MG and Acampora, AS, "The Knockout Switch: A Simple, Modular Architecture for High-Performance Packet Switching", IEEE Journal on Selected Areas in Communications, Vol 5, No 8, Oct 1987, pgs 1274-1283.

Appendix A - Sample Simulation Input and Output Files

The simulation program is written in the C programming language, and uses a make file to coordinate the compiling of the various modules. The code has not been reproduced here for space considerations. The modules and their purpose are:

mk	UNIX shell wrapper to invoke make properly for the desired platform.
control.make	the makefile used to compile the modules.
borland.c	UNIX functions that are not quite the same on a PC
config.h, config.c	reads the switch configuration in from the input file and writes the switch configuration entries to the output files.
control.h, control.c	the main function, shows revisions, program operation, and other information.
controlr.h, controlr.c	the controller specific functions for the different switch architectures and sequence preserving algorithms.
cpu_spec.h	defines for the various CPUs the program can be compiled for.
debug.h	defines for enabling various debug logging.
defines.h	global defines for maximum values, cell structures.
files.h, files.c	functions to open, close, and flush the files used in the simulation.
signals.h, signals.c	receive signals directed to the process, for shutting the simulation down nicely.
simulate.h, simulate.c	controls the actual simulation, routing the ATM cells to the outlets.
source.h, source.c	generating the randomly distributed ATM cells that arrive at each input.
state.h, state.c	functions to save and restore the state of the simulation to a file (used to recover in the event of a system shutdown during a simulation run).
stats.h, stats.c	functions to clear and gather the various statistics about cell loss probability, queue usage, etc.
time.h, time.c	functions used to time the simulation runs.

The first file shown on the following pages, 'cfg.', is the input configuration file showing the switch architecture to be simulated. The configuration and results files are for a simple reflex switch architecture with the switch core operating at twice the packet rate of the inputs. The switch has 8 inputs, 8 outputs, and 8 single packet buffers for a total of 16 inlets and 16

outlets. The switch was simulated for 10 offered traffic loads of 1 million packets each. Each of these runs of 1 million packets was broken into 10 separate runs of 100,000 packets each. To prevent the simulation from starting with empty buffers, the simulation runs through 20 times the number of inputs times the total number of buffer spaces within the switch time slots before starting to gather statistics. The next five files show the results of the simulation run, with each file showing the following:

'reflext2.'	running time and other general information.
'reflext2.hist'	histogram of packet latency.
'reflext2.input'	packet loss probability on a per input basis.
'reflext2.knock'	packet loss probability on a per batch basis.
'reflext2.output'	packet loss probability on a per output basis.

File 'cfg.'

```
reflex fastt2
preserveio
orderbi
#inputs : 8
#outputs : 8
#buffers : 8
1 1 1 1 1 1 1 1
#loads : 10
0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0
#cells : 1e6
#batchs : 10
file : reflext2
```

File 'reflext2.'

```
Simulation started on Rob's PC at Sun Apr 10 01:42:50 1994
Timing : simulation started at 765956570.810
PROGRESS : Max_Delay = -1, Max_Loops = -1
Progress : Load 0.100 : batchs 0  1  2  3  4  5  6  7  8  9  done - took 0 days 0:27
Progress : Load 0.200 : batchs 0  1  2  3  4  5  6  7  8  9  done - took 0 days 0:01
Progress : Load 0.300 : batchs 0  1  2  3  4  5  6  7  8  9  done - took 0 days 0:00
Progress : Load 0.400 : batchs 0  1  2  3  4  5  6  7  8  9  done - took 0 days 0:00
Progress : Load 0.500 : batchs 0  1  2  3  4  5  6  7  8  9  done - took 0 days 0:00
Progress : Load 0.600 : batchs 0  1  2  3  4  5  6  7  8  9  done - took 0 days 0:00
Progress : Load 0.700 : batchs 0  1  2  3  4  5  6  7  8  9  done - took 0 days 0:00
Progress : Load 0.800 : batchs 0  1  2  3  4  5  6  7  8  9  done - took 0 days 0:00
Progress : Load 0.900 : batchs 0  1  2  3  4  5  6  7  8  9  done - took 0 days 0:00
Progress : Load 1.000 : batchs 0  1  2  3  4  5  6  7  8  9  done - took 0 days 0:00
Timing : simulation ended at 765958375.550
Timing : total simulation took 0 days 0:30
Simulation ended on Rob's PC at Sun Apr 10 02:12:55 1994
```

File 'reflext2.hist'

Reflex Switch Controller (Rob's PC at Sun Apr 10 01:42:50 1994)
Order on Input and Output Cell Order
Orderbi Priority Input Priority
1.000e+06 cells (10 batchs) sent into switch configuration :
8 inputs, 8 buffers, 8 outputs
10 Loads to simulate : 0.1 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1
8 Buffers (max delay is 1) :
Buffer 0 : -- -- --
Buffer 1 : -- -- --
Buffer 2 : -- -- --
Buffer 3 : -- -- --
Buffer 4 : -- -- --
Buffer 5 : -- -- --
Buffer 6 : -- -- --
Buffer 7 : -- -- --

Load	Max	0	1	2	3	4	5	6	7	8
0.1	7	593488	282597	90999	21638	4051	518	55	2	0
	7	593488	282597	90999	21638	4051	518	55	2	0
0.2	8	10257	14894	16480	13601	8191	3450	864	136	8
	8	10257	14894	16480	13601	8191	3450	864	136	8
0.3	8	124	533	1531	3105	4407	4104	2310	713	94
	8	124	533	1531	3105	4407	4104	2310	713	94
0.4	8	1	13	94	407	1272	2452	2699	1533	357
	8	1	13	94	407	1272	2452	2699	1533	357
0.5	8	1	1	7	61	271	1065	2095	1891	624
	8	1	1	7	61	271	1065	2095	1891	624
0.6	8	1	1	3	6	46	370	1295	1856	949
	8	1	1	3	6	46	370	1295	1856	949
0.7	8	1	1	1	1	11	115	726	1701	1143
	8	1	1	1	1	11	115	726	1701	1143
0.8	8	1	1	1	1	2	26	365	1327	1350
	8	1	1	1	1	2	26	365	1327	1350
0.9	8	1	1	1	1	1	5	109	1034	1479
	8	1	1	1	1	1	5	109	1034	1479
1.0	8	1	1	1	1	1	1	2	769	1534
	8	1	1	1	1	1	1	2	769	1534

File 'reflext2.input'

```
Reflex Switch Controller (Rob's PC at Sun Apr 10 01:42:50 1994)
Order on Input and Output Cell Order
Orderbi Priority Input Priority
1.000e+06 cells (10 batchs) sent into switch configuration :
8 inputs, 8 buffers, 8 outputs
10 Loads to simulate :   0.1   0.2   0.3   0.4   0.5   0.6   0.7   0.8   0.9   1
8 Buffers (max delay is 1) :
Buffer  0 : -- -- --
Buffer  1 : -- -- --
Buffer  2 : -- -- --
Buffer  3 : -- -- --
Buffer  4 : -- -- --
Buffer  5 : -- -- --
Buffer  6 : -- -- --
Buffer  7 : -- -- --
```

utili	start	end	into	through	blocking	collision	cell loss	out of seqo	out of seqoi	avg delay	avg buffered
0.1	0	0	125376	125133	243	0	243	0	0	27255	27255
0.1	1	2	124890	124613	276	0	276	0	0	39334	39334
0.1	1	2	124203	123781	421	0	421	0	0	50743	50743
0.1	1	1	124221	123615	606	0	606	0	0	62612	62612
0.1	2	2	125731	124883	848	0	848	0	0	75437	75437
0.1	0	0	124999	123856	1143	0	1143	0	0	86403	86403
0.1	1	2	125205	123831	1373	0	1373	0	0	97391	97391
0.1	1	1	125389	123636	1753	0	1753	0	0	109472	109472
0.2	4	3	9783	8946	838	0	838	0	0	13956	13956
0.2	2	1	9503	8587	917	0	917	0	0	15153	15153
0.2	5	5	9926	8839	1087	0	1087	0	0	16812	16812
0.2	5	5	9519	8297	1222	0	1222	0	0	17374	17374
0.2	4	4	9808	8486	1322	0	1322	0	0	18965	18965
0.2	5	5	9820	8408	1412	0	1412	0	0	20034	20034
0.2	6	6	9883	8310	1573	0	1573	0	0	20948	20948
0.2	7	7	9638	8008	1630	0	1630	0	0	21629	21629
0.3	1	2	3376	2296	1079	0	1079	0	0	8183	8183
0.3	7	8	3413	2265	1147	0	1147	0	0	8469	8469
0.3	5	6	3348	2162	1185	0	1185	0	0	8542	8542
0.3	4	5	3372	2132	1239	0	1239	0	0	8735	8735
0.3	3	4	3353	2058	1294	0	1294	0	0	8886	8886
0.3	7	8	3343	2043	1299	0	1299	0	0	9184	9184
0.3	7	8	3365	2012	1352	0	1352	0	0	9242	9242
0.3	5	6	3362	1953	1408	0	1408	0	0	9420	9420
0.4	7	8	2378	1223	1154	0	1154	0	0	5933	5933
0.4	4	5	2411	1170	1240	0	1240	0	0	5976	5976
0.4	7	8	2340	1135	1204	0	1204	0	0	6000	6000
0.4	7	8	2349	1101	1247	0	1247	0	0	6039	6039
0.4	5	5	2305	1058	1247	0	1247	0	0	5955	5955
0.4	5	6	2356	1082	1273	0	1273	0	0	6223	6223
0.4	5	6	2304	1023	1280	0	1280	0	0	6095	6095
0.4	8	9	2397	1036	1360	0	1360	0	0	6330	6330
0.5	8	8	2024	827	1197	0	1197	0	0	4642	4642
0.5	6	7	2021	795	1225	0	1225	0	0	4606	4606
0.5	6	7	2001	782	1218	0	1218	0	0	4692	4692
0.5	7	8	1993	754	1238	0	1238	0	0	4680	4680
0.5	8	9	1979	739	1239	0	1239	0	0	4672	4672

0.5	8	9	2008	719	1288	0	1288	0	0	4698	4698
0.5	6	7	1950	690	1259	0	1259	0	0	4591	4591
0.5	9	9	2056	710	1346	0	1346	0	0	4825	4825
0.6	8	9	1846	607	1238	0	1238	0	0	3754	3754
0.6	9	10	1792	595	1196	0	1196	0	0	3782	3782
0.6	8	9	1841	583	1257	0	1257	0	0	3790	3790
0.6	8	9	1781	574	1206	0	1206	0	0	3807	3807
0.6	9	10	1841	562	1278	0	1278	0	0	3821	3821
0.6	9	10	1842	550	1291	0	1291	0	0	3853	3853
0.6	9	10	1821	536	1284	0	1284	0	0	3824	3824
0.6	8	9	1783	520	1262	0	1262	0	0	3782	3782
0.7	7	8	1724	497	1226	0	1226	0	0	3258	3258
0.7	7	8	1770	488	1281	0	1281	0	0	3259	3259
0.7	8	9	1726	475	1250	0	1250	0	0	3240	3240
0.7	8	9	1665	463	1201	0	1201	0	0	3217	3217
0.7	8	9	1665	450	1214	0	1214	0	0	3219	3219
0.7	9	10	1716	453	1262	0	1262	0	0	3296	3296
0.7	9	10	1696	436	1259	0	1259	0	0	3246	3246
0.7	9	10	1760	438	1321	0	1321	0	0	3297	3297
0.8	6	7	1656	411	1244	0	1244	0	0	2833	2833
0.8	9	10	1594	407	1186	0	1186	0	0	2821	2821
0.8	9	10	1620	399	1220	0	1220	0	0	2833	2833
0.8	9	10	1646	392	1253	0	1253	0	0	2838	2838
0.8	9	10	1636	381	1254	0	1254	0	0	2807	2807
0.8	8	9	1659	373	1285	0	1285	0	0	2805	2805
0.8	7	8	1664	360	1303	0	1303	0	0	2754	2754
0.8	9	10	1628	351	1276	0	1276	0	0	2732	2732
0.9	9	10	1588	352	1235	0	1235	0	0	2526	2526
0.9	9	10	1585	350	1234	0	1234	0	0	2538	2538
0.9	7	8	1591	344	1246	0	1246	0	0	2533	2533
0.9	9	10	1571	338	1232	0	1232	0	0	2509	2509
0.9	9	10	1586	333	1252	0	1252	0	0	2494	2494
0.9	9	10	1602	315	1286	0	1286	0	0	2430	2430
0.9	9	10	1574	302	1271	0	1271	0	0	2373	2373
0.9	9	10	1565	298	1266	0	1266	0	0	2356	2356
1	9	10	1541	309	1231	0	1231	0	0	2308	2308
1	9	10	1541	308	1232	0	1232	0	0	2303	2303
1	9	10	1541	308	1232	0	1232	0	0	2305	2305
1	9	10	1541	308	1232	0	1232	0	0	2306	2306
1	9	10	1541	308	1232	0	1232	0	0	2307	2307
1	9	10	1541	257	1283	0	1283	0	0	2052	2052
1	9	10	1541	257	1283	0	1283	0	0	2054	2054
1	9	10	1541	256	1284	0	1284	0	0	2047	2047

File 'reflext2.knock'

```
Reflex Switch Controller (Rob's PC at Sun Apr 10 01:42:50 1994)
Order on Input and Output Cell Order
Orderbi Priority Input Priority
1.000e+06 cells (10 batchs) sent into switch configuration :
8 inputs, 8 buffers, 8 outputs
10 Loads to simulate :  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1
8 Buffers (max delay is 1) :
Buffer  0 : -- -- --
Buffer  1 : -- -- --
Buffer  2 : -- -- --
Buffer  3 : -- -- --
Buffer  4 : -- -- --
Buffer  5 : -- -- --
Buffer  6 : -- -- --
Buffer  7 : -- -- --
```

utili	start	end	into	through	block-ing	col-lision	cell loss	out of seqo	out of seqoi	avg delay	avg buffered
0.1	0	1	100002	99276	725	0	725	0	0	54755	54755
0.1	1	3	100002	99364	636	0	636	0	0	54718	54718
0.1	3	2	100002	99334	669	0	669	0	0	54585	54585
0.1	2	0	100001	99315	688	0	688	0	0	54456	54456
0.1	0	0	100001	99346	655	0	655	0	0	54737	54737
0.1	0	0	100001	99358	643	0	643	0	0	55253	55253
0.1	0	0	100001	99321	680	0	680	0	0	54837	54837
0.1	0	0	100001	99358	643	0	643	0	0	54585	54585
0.1	0	1	100002	99345	656	0	656	0	0	54978	54978
0.1	1	3	100001	99331	668	0	668	0	0	55743	55743
0.2	6	3	7362	6365	1000	0	1000	0	0	13793	13793
0.2	3	5	7224	6221	1001	0	1001	0	0	13656	13656
0.2	5	2	7468	6471	1000	0	1000	0	0	14316	14316
0.2	2	5	7912	6909	1000	0	1000	0	0	14705	14705
0.2	5	2	7437	6440	1000	0	1000	0	0	14216	14216
0.2	2	3	8928	7927	1000	0	1000	0	0	15850	15850
0.2	3	4	8080	7079	1000	0	1000	0	0	14703	14703
0.2	4	5	8320	7319	1000	0	1000	0	0	15019	15019
0.2	5	3	7481	6483	1000	0	1000	0	0	14181	14181
0.2	3	4	7668	6667	1000	0	1000	0	0	14432	14432
0.3	0	5	2682	1677	1000	0	1000	0	0	7052	7052
0.3	5	7	2728	1725	1001	0	1001	0	0	7204	7204
0.3	7	3	2802	1805	1001	0	1001	0	0	7299	7299
0.3	3	5	2751	1749	1000	0	1000	0	0	7345	7345
0.3	5	3	2572	1573	1001	0	1001	0	0	6760	6760
0.3	3	4	2753	1752	1000	0	1000	0	0	7145	7145
0.3	4	4	2618	1618	1000	0	1000	0	0	6990	6990
0.3	4	3	2592	1593	1000	0	1000	0	0	6810	6810
0.3	3	5	2731	1729	1000	0	1000	0	0	7189	7189
0.3	5	8	2703	1700	1000	0	1000	0	0	6867	6867
0.4	0	6	1894	887	1001	0	1001	0	0	4893	4893
0.4	6	4	1877	879	1000	0	1000	0	0	4853	4853
0.4	4	4	1865	863	1002	0	1002	0	0	4804	4804
0.4	4	4	1891	891	1000	0	1000	0	0	4720	4720
0.4	4	6	1869	867	1000	0	1000	0	0	4781	4781
0.4	6	7	1918	916	1001	0	1001	0	0	5019	5019
0.4	7	7	1895	894	1001	0	1001	0	0	4940	4940

0.4	7	5	1843	845	1000	0	1000	0	0	4759	4759
0.4	5	5	1921	921	1000	0	1000	0	0	4976	4976
0.4	5	7	1867	865	1000	0	1000	0	0	4806	4806
0.5	0	6	1622	616	1000	0	1000	0	0	3827	3827
0.5	6	7	1607	606	1000	0	1000	0	0	3758	3758
0.5	7	6	1607	607	1001	0	1001	0	0	3789	3789
0.5	6	7	1581	579	1001	0	1001	0	0	3643	3643
0.5	7	8	1612	611	1000	0	1000	0	0	3751	3751
0.5	8	5	1595	595	1003	0	1003	0	0	3715	3715
0.5	5	5	1608	605	1003	0	1003	0	0	3769	3769
0.5	5	7	1627	625	1000	0	1000	0	0	3796	3796
0.5	7	7	1595	594	1001	0	1001	0	0	3698	3698
0.5	7	6	1578	578	1001	0	1001	0	0	3660	3660
0.6	0	5	1459	454	1000	0	1000	0	0	3059	3059
0.6	5	8	1464	458	1003	0	1003	0	0	3024	3024
0.6	8	8	1444	444	1000	0	1000	0	0	3039	3039
0.6	8	8	1457	457	1000	0	1000	0	0	3077	3077
0.6	8	8	1451	449	1002	0	1002	0	0	2977	2977
0.6	8	7	1455	456	1000	0	1000	0	0	3082	3082
0.6	7	8	1449	445	1003	0	1003	0	0	2993	2993
0.6	8	8	1450	449	1001	0	1001	0	0	3010	3010
0.6	8	8	1448	448	1000	0	1000	0	0	3007	3007
0.6	8	8	1470	467	1003	0	1003	0	0	3145	3145
0.7	0	7	1377	366	1004	0	1004	0	0	2561	2561
0.7	7	7	1376	376	1000	0	1000	0	0	2658	2658
0.7	7	7	1360	359	1001	0	1001	0	0	2535	2535
0.7	7	7	1373	372	1001	0	1001	0	0	2629	2629
0.7	7	8	1363	361	1001	0	1001	0	0	2554	2554
0.7	8	7	1374	375	1000	0	1000	0	0	2638	2638
0.7	7	7	1365	365	1000	0	1000	0	0	2573	2573
0.7	7	8	1381	379	1001	0	1001	0	0	2622	2622
0.7	8	7	1372	371	1002	0	1002	0	0	2629	2629
0.7	7	8	1381	376	1004	0	1004	0	0	2633	2633
0.8	0	7	1322	313	1002	0	1002	0	0	2263	2263
0.8	7	7	1309	307	1002	0	1002	0	0	2256	2256
0.8	7	8	1313	309	1003	0	1003	0	0	2219	2219
0.8	8	7	1301	302	1000	0	1000	0	0	2217	2217
0.8	7	7	1310	309	1001	0	1001	0	0	2259	2259
0.8	7	8	1309	304	1004	0	1004	0	0	2221	2221
0.8	8	7	1302	303	1000	0	1000	0	0	2237	2237
0.8	7	8	1310	308	1001	0	1001	0	0	2243	2243
0.8	8	7	1315	312	1004	0	1004	0	0	2271	2271
0.8	7	8	1312	307	1004	0	1004	0	0	2237	2237
0.9	0	8	1282	269	1005	0	1005	0	0	1987	1987
0.9	8	8	1259	258	1001	0	1001	0	0	1945	1945
0.9	8	7	1266	263	1004	0	1004	0	0	1979	1979
0.9	7	8	1263	262	1000	0	1000	0	0	1981	1981
0.9	8	8	1264	264	1000	0	1000	0	0	1980	1980
0.9	8	8	1265	264	1001	0	1001	0	0	1986	1986
0.9	8	7	1265	261	1005	0	1005	0	0	1971	1971
0.9	7	8	1269	264	1004	0	1004	0	0	1969	1969
0.9	8	8	1266	264	1002	0	1002	0	0	1988	1988
0.9	8	8	1263	263	1000	0	1000	0	0	1973	1973
1	0	8	1240	232	1000	0	1000	0	0	1743	1743
1	8	8	1232	231	1001	0	1001	0	0	1771	1771
1	8	8	1232	231	1001	0	1001	0	0	1771	1771
1	8	8	1232	231	1001	0	1001	0	0	1771	1771
1	8	8	1232	231	1001	0	1001	0	0	1771	1771

1	8	8	1232	231	1001	0	1001	0	0	1771	1771
1	8	8	1232	231	1001	0	1001	0	0	1771	1771
1	8	8	1232	231	1001	0	1001	0	0	1771	1771
1	8	8	1232	231	1001	0	1001	0	0	1771	1771
1	8	8	1232	231	1001	0	1001	0	0	1771	1771

File 'reflext2.output'

```
Reflex Switch Controller (Rob's PC at Sun Apr 10 01:42:50 1994)
Order on Input and Output Cell Order
Orderbi Priority Input Priority
1.000e+06 cells (10 batchs) sent into switch configuration :
8 inputs, 8 buffers, 8 outputs
10 Loads to simulate :  0.1  0.2  0.3  0.4  0.5  0.6  0.7  0.8  0.9  1
8 Buffers (max delay is 1) :
Buffer  0 : -- -- --
Buffer  1 : -- -- --
Buffer  2 : -- -- --
Buffer  3 : -- -- --
Buffer  4 : -- -- --
Buffer  5 : -- -- --
Buffer  6 : -- -- --
Buffer  7 : -- -- --
```

utili	start	end	into	through	block-ing	col-lision	cell loss	out of seqo	out of seqoi	avg delay	avg buffered
0.1	7	10	1000014	993348	6663	0	6663	0	0	548647	548647
0.1	0	0	0	0	0	0	0	0	0	0	0
0.1	0	0	0	0	0	0	0	0	0	0	0
0.1	0	0	0	0	0	0	0	0	0	0	0
0.1	0	0	0	0	0	0	0	0	0	0	0
0.1	0	0	0	0	0	0	0	0	0	0	0
0.1	0	0	0	0	0	0	0	0	0	0	0
0.1	0	0	0	0	0	0	0	0	0	0	0
0.2	38	36	77880	67881	10001	0	10001	0	0	144871	144871
0.2	0	0	0	0	0	0	0	0	0	0	0
0.2	0	0	0	0	0	0	0	0	0	0	0
0.2	0	0	0	0	0	0	0	0	0	0	0
0.2	0	0	0	0	0	0	0	0	0	0	0
0.2	0	0	0	0	0	0	0	0	0	0	0
0.2	0	0	0	0	0	0	0	0	0	0	0
0.2	0	0	0	0	0	0	0	0	0	0	0
0.3	39	47	26932	16921	10003	0	10003	0	0	70661	70661
0.3	0	0	0	0	0	0	0	0	0	0	0
0.3	0	0	0	0	0	0	0	0	0	0	0
0.3	0	0	0	0	0	0	0	0	0	0	0
0.3	0	0	0	0	0	0	0	0	0	0	0
0.3	0	0	0	0	0	0	0	0	0	0	0
0.3	0	0	0	0	0	0	0	0	0	0	0
0.3	0	0	0	0	0	0	0	0	0	0	0
0.4	48	55	18840	8828	10005	0	10005	0	0	48551	48551
0.4	0	0	0	0	0	0	0	0	0	0	0
0.4	0	0	0	0	0	0	0	0	0	0	0
0.4	0	0	0	0	0	0	0	0	0	0	0
0.4	0	0	0	0	0	0	0	0	0	0	0
0.4	0	0	0	0	0	0	0	0	0	0	0
0.4	0	0	0	0	0	0	0	0	0	0	0
0.4	0	0	0	0	0	0	0	0	0	0	0
0.5	58	64	16032	6016	10010	0	10010	0	0	37406	37406
0.5	0	0	0	0	0	0	0	0	0	0	0
0.5	0	0	0	0	0	0	0	0	0	0	0
0.5	0	0	0	0	0	0	0	0	0	0	0
0.5	0	0	0	0	0	0	0	0	0	0	0

0.5	0	0	0	0	0	0	0	0	0	0	0
0.5	0	0	0	0	0	0	0	0	0	0	0
0.5	0	0	0	0	0	0	0	0	0	0	0
0.6	68	76	14547	4527	10012	0	10012	0	0	30413	30413
0.6	0	0	0	0	0	0	0	0	0	0	0
0.6	0	0	0	0	0	0	0	0	0	0	0
0.6	0	0	0	0	0	0	0	0	0	0	0
0.6	0	0	0	0	0	0	0	0	0	0	0
0.6	0	0	0	0	0	0	0	0	0	0	0
0.6	0	0	0	0	0	0	0	0	0	0	0
0.6	0	0	0	0	0	0	0	0	0	0	0
0.7	65	73	13722	3700	10014	0	10014	0	0	26032	26032
0.7	0	0	0	0	0	0	0	0	0	0	0
0.7	0	0	0	0	0	0	0	0	0	0	0
0.7	0	0	0	0	0	0	0	0	0	0	0
0.7	0	0	0	0	0	0	0	0	0	0	0
0.7	0	0	0	0	0	0	0	0	0	0	0
0.7	0	0	0	0	0	0	0	0	0	0	0
0.7	0	0	0	0	0	0	0	0	0	0	0
0.8	66	74	13103	3074	10021	0	10021	0	0	22423	22423
0.8	0	0	0	0	0	0	0	0	0	0	0
0.8	0	0	0	0	0	0	0	0	0	0	0
0.8	0	0	0	0	0	0	0	0	0	0	0
0.8	0	0	0	0	0	0	0	0	0	0	0
0.8	0	0	0	0	0	0	0	0	0	0	0
0.8	0	0	0	0	0	0	0	0	0	0	0
0.8	0	0	0	0	0	0	0	0	0	0	0
0.9	70	78	12662	2632	10022	0	10022	0	0	19759	19759
0.9	0	0	0	0	0	0	0	0	0	0	0
0.9	0	0	0	0	0	0	0	0	0	0	0
0.9	0	0	0	0	0	0	0	0	0	0	0
0.9	0	0	0	0	0	0	0	0	0	0	0
0.9	0	0	0	0	0	0	0	0	0	0	0
0.9	0	0	0	0	0	0	0	0	0	0	0
0.9	0	0	0	0	0	0	0	0	0	0	0
1	72	80	12328	2311	10009	0	10009	0	0	17682	17682
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	0	0	0	0	0	0	0

Appendix B - Crossbar Switch Theoretical and Simulated Performance

Assume a crossbar switch with N inputs, M outputs, and a probability of a packet arriving at input i in a single time slot of ρ_i as shown in figure B.1 Also assume the probability of a packet arriving at input i being destined for output j is given by $\sigma_i(j)$. This results in the probability of a packet arriving at an input in a single time slot destined for a specific output being given by:

$$P(\textit{output } j \textit{ has a packet arrive from input } i) = \rho_i \cdot \sigma_i(j) \qquad \text{Eq. B.1}$$

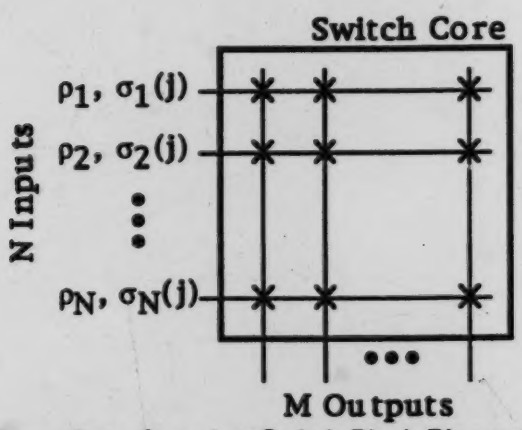


Figure B.1 - Crossbar Switch Block Diagram

The probability of output j having one or more packets arrive at all the N inputs is given by equation B.2, while the probability of output j having exactly 1 packet arrive at any of the N inputs is given by equation B.3. Equation B.4 gives the probability of output j having no packets arrive at any input.

$$P(\textit{output } j \textit{ has one or more packets arrive from any input}) = \sum_{i=1}^{N} \rho_i \cdot \sigma_i(j) \qquad \text{Eq. B.2}$$

$$P(\textit{output } j \textit{ has one packet arrive from any input}) = \sum_{i=1}^{N} \rho_i \cdot \sigma_i(j) \cdot \prod_{\substack{k=1 \\ k \neq i}}^{N} (1 - \rho_k \cdot \sigma_k(j)) \qquad \text{Eq. B.3}$$

$$P(\textit{output } j \textit{ has zero packets arrive from any input}) = \prod_{i=1}^{N} (1 - \rho_i \cdot \sigma_i(j)) \qquad \text{Eq. B.4}$$

The probability of a packet being lost is calculated by dividing the packet

throughput of output j by the packet arrival rate for output j as shown in equation B.5 and expanded in equation B.6.

$$PLP_{output\ j} = \frac{P(\underset{packet(s)\ arrive}{output\ j\ has}) - P(\underset{a\ packet\ out}{output\ j\ sends})}{P(\underset{packet(s)\ arrive}{output\ j\ has})}$$

$$= \frac{P(\underset{packet(s)\ arrive}{output\ j\ has}) - \left[1 - P(\underset{arrive\ from\ any\ input}{output\ j\ has\ zero\ packets})\right]}{P(\underset{packet(s)\ arrive}{output\ j\ has})} \qquad \text{Eq. B.5}$$

$$PLP_{output\ j} = \frac{\sum_{i=1}^{N} \rho_i \cdot \sigma_i(j) - \left[1 - \prod_{i=1}^{N} (1 - \rho_i \cdot \sigma_i(j))\right]}{\sum_{i=1}^{N} \rho_i \cdot \sigma_i(j))} \qquad \text{Eq. B.6}$$

To match the simulation all the traffic load probability functions were assumed to be uniform with the packet destinations spread uniformly over all the outputs and the same for all inputs, reducing the above equation to :

$$\rho_i = \rho \qquad for\ all\ i = 1\ to\ N\ inputs$$

$$\sigma_i(j) = \frac{1}{M} \qquad for\ all\ i = 1\ to\ N\ inputs,\ j = 1\ to\ M\ outputs \qquad \text{Eq. B.7}$$

$$PLP_{output\ j} = \frac{\frac{N \cdot \rho}{M} - \left[1 - \left(1 - \frac{\rho}{M}\right)^N\right]}{\frac{N \cdot \rho}{M}} = 1 - \frac{M}{N \cdot \rho} \cdot \left[1 - \left(1 - \frac{\rho}{M}\right)^N\right]$$

Since this result applies to any of the outputs, and all outputs are treated equally within the crossbar switch, this result applies to the switch as a whole. Thus the packet loss probability of the entire switch is given by equation B.8.

$$PLP = 1 - \frac{M}{N \cdot \rho} \cdot \left[1 - \left(1 - \frac{\rho}{M}\right)^N\right] \qquad \text{Eq. B.8}$$

Table B.1 shows the summarized output of a set of nine simulation runs sending 1 billion packets in ten batches through an 8 input by 8 output crossbar switch. The data shows the theoretical calculations using equation B.8 to match the simulated results very closely. Assuming a normal distribution of the packet

loss probability among the ten batches is not proper due to the limited number of batches. The student's t-curve is a better approximation, depending on converting the calculated standard deviation to a slightly larger value as shown in equation B.9. The 95 % confidence interval is then given by the standard deviation times 2.26 for the ten measurements rather than the factor of 2 used for the normal curve [Fre78].

$$StdDev^{+} = \frac{Number\ Measurements}{Number\ Measurements - 1} \cdot [StdDev] \qquad \text{Eq. B.9}$$

Crosspoint
Random
Orderbi
1e+09 cells
10 batchs
from xpt8_8
8x1x8
loads 9

Load	CLP Theory	CLP Batchs	Std Dev	Std Dev+	95% CI	95% low	95% high
0.2	0.083259	0.083249	2.34294E-6	2.6033E-06	5.8834E-06	0.083244	0.083255
0.4	0.158551	0.158551	2.1611E-6	2.4012E-06	5.4268E-06	0.158546	0.158556
0.6	0.226603	0.226607	2.10178E-6	2.3353E-06	5.2778E-06	0.226602	0.226612
0.7	0.258125	0.258125	2.04332E-6	2.2704E-06	5.131E-06	0.258120	0.258130
0.8	0.288084	0.288081	4.0184E-6	4.4649E-06	1.0091E-05	0.288071	0.288091
0.9	0.316555	0.316566	3.03492E-6	3.3721E-06	7.621E-06	0.316558	0.316574
0.95	0.330254	0.330257	2.74824E-6	3.0536E-06	6.9011E-06	0.330250	0.330264
0.99	0.340965	0.340963	1.94868E-6	2.1652E-06	4.8934E-06	0.340958	0.340968
1	0.343609	0.343612	4.13079E-7	4.5898E-07	1.0373E-06	0.343611	0.343613

Table B.1 - Crossbar Switch PLP - Theory and Simulated per Batch

Table B.2 shows similar results collected on the same simulation runs averaging across the eight individual outputs rather than for the ten batches. Because of the fewer measurements, the student's t-curve factor is increased to 2.37 from 2.26. Again, the simulations agree very well with the theoretical results. The graph in figure B.2 shows the theoretical packet loss probability minus the simulated average packet loss probability showing the 95 % confidence interval bars. Note that the vertical scale is measured in 10^{-6} units.

Crosspoint
Random
Orderbi
1e+09 cells
10 batchs
from xpt8_8
8x1x8
loads 9

Load	CLP Theory	CLP Outputs	Std Dev	Std Dev+	95% CI	95% low	95% high
0.2	0.083259	0.083245	2.5718E-06	2.9392E-06	6.9659E-06	0.083242	0.083256
0.4	0.158551	0.158551	2.2981E-06	2.6263E-06	6.2244E-06	0.158545	0.158557
0.6	0.226603	0.226607	3.5911E-06	4.1041E-06	9.7266E-06	0.226597	0.226617
0.7	0.258125	0.258125	1.1922E-06	1.3625E-06	3.2291E-06	0.258121	0.258128
0.8	0.288084	0.288081	2.5099E-06	2.8684E-06	6.7981E-06	0.288074	0.288087
0.9	0.316555	0.316566	1.7044E-06	1.9479E-06	4.6165E-06	0.316561	0.316571
0.95	0.330254	0.330257	2.3416E-06	2.6761E-06	6.3423E-06	0.330251	0.330263
0.99	0.340965	0.340963	3.2638E-06	3.7301E-06	8.8403E-06	0.340954	0.340972
1	0.343609	0.343612	5.0331E-07	5.7521E-07	1.3632E-06	0.343611	0.343613

Table B.2 - Crossbar Switch PLP - Theory and Simulated per Output

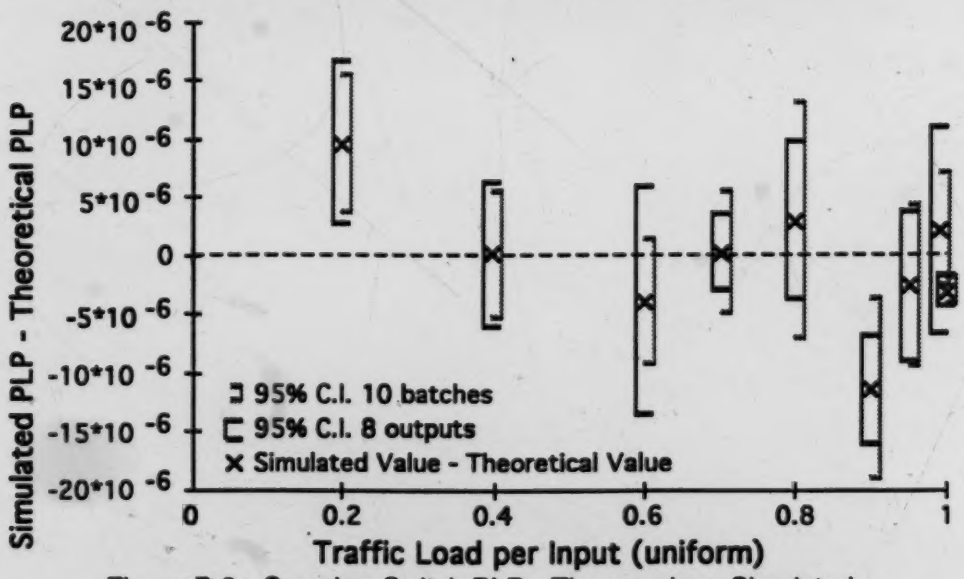


Figure B.2 - Crossbar Switch PLP - Theory minus Simulated

Figure B.2 shows that the simulated results match the theoretical results within the 95 % confidence interval in all but the 0.2 and 0.95 traffic load simulation runs. Those two are just slightly out of the 95 % confidence interval calculated.

Appendix C - Knockout Switch Theoretical Performance

Assume a knockout switch with N inputs, M outputs, L paths to each output, and a probability of a packet arriving at input i in a single time slot of ρ_i as shown in figure C.1 Also assume that the probability of a packet arriving at input i being destined for output j is given by $\sigma_i(j)$. This results in the probability of a packet arriving at an input in a single time slot destined for a specific output being given by:

$$P(\text{output } j \text{ has a packet arrive from input } i) = \rho_i \cdot \sigma_i(j) \qquad \text{Eq. C.1}$$

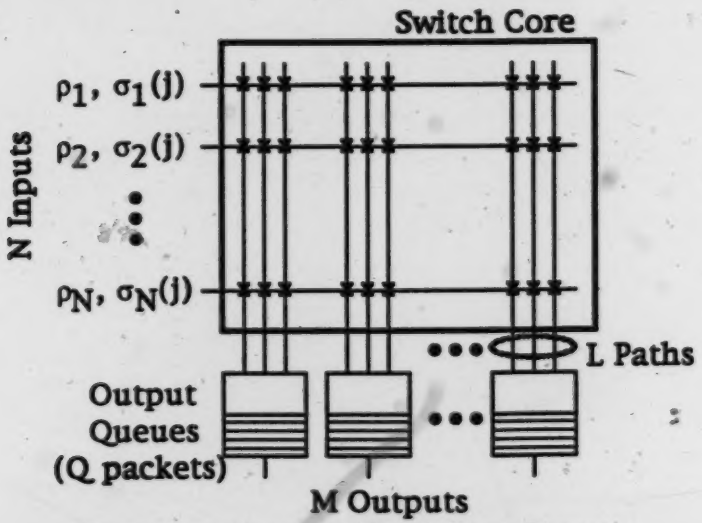


Figure C.1 - Knockout Switch Block Diagram

As discussed in the text, there are two distinct causes of packet loss in the knockout switch: the collision of more than L packets at a specific output; and the overflow of the simple output queue at a specific output. The collision of more than L packets will be examined first. Since equations B.2, B.3, and B.4 all apply to the knockout switch as well as the simple crossbar switch, they show how the probability of exactly i packets arriving at a single output j. The probability of a packet arriving at an output being lost due to packets colliding at a single output is the probability of more than L packets arriving at all the inputs

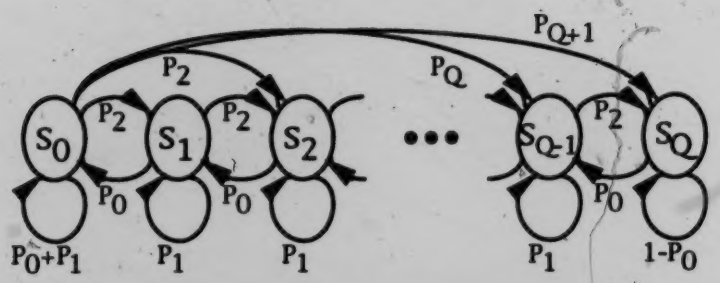
destined for a single output j in a single time slot times the number of packets in excess of L as shown in equation C.2. The packet loss probability is this probability scaled by the probability of any packet arriving at any input destined for this output.

$$PLP_{collision\ at\ output\ j} = \frac{\sum_{i=L+1}^{N} (i-L)\cdot P(\substack{output\ j\ has\ exactly \\ i\ packets\ arrive})}{\frac{N\cdot\rho}{M}} \qquad \text{Eq. C.2}$$

The probability of exactly i packets arriving destined for a specific output j is given by equation C.3, again assuming all the traffic load probability functions are uniform with the packet destinations spread uniformly over all the outputs.

$$\begin{aligned} &\rho_i = \rho \qquad for\ all\ i = 1\ to\ N\ inputs \\ &\sigma_i(j) = \frac{1}{M} \quad for\ all\ i = 1\ to\ N\ inputs,\ j = 1\ to\ M\ outputs \\ &P_i = C_i^N \cdot \left(\frac{\rho}{M}\right)^i \cdot \left(1-\frac{\rho}{M}\right)^{N-i} \quad for\ i = 0,1,\ldots,L \\ &\quad = 0 \qquad\qquad\qquad\qquad otherwise \end{aligned} \qquad \text{Eq. C.3}$$

To determine the probability of a packet being lost due to queue overflow at a specific output, the state transition probabilities of the output queue must be determined [Kar87][Hlu88]. For a simple output queue that holds Q packet, there are Q+1 possible states (S_0 to S_Q) ranging from the queue being completely empty to completely full. The state transition diagram is shown in figure C.2 with all multi-state transitions removed for all except the S_0 state for clarity. As this diagram shows, if no packets arrive (P_0) the queue empties by one packet unless the queue is already empty. If exactly one packet (P_1) arrives, the queue stays the same. If exactly L packets arrive, the queue fills up by L-1 packets unless this would exceed the maximum that the queue can hold.



Multi-state transitions from other than S_0 are not shown for clarity.

Figure C.2 - Knockout Switch State Transition Diagram

Equation C.4 shows the transition probabilities as derived from figure C.2. To change from state i to state i+1 requires exactly two packets to arrive. To stay in state i requires exactly one packet to arrive. If no packets arrive, the state changes from i to i-1, which is the only way for the queue to empty. The special case of the empty queue state (S_0) does not decrease if no packets arrive. Likewise, the full queue state (S_Q) is the upper bound on the queue states that cannot be passed. This leads to the transition probabilities $P_{si->sj}$ (changing from state i to state j) shown in equation C.4.

$$
\begin{aligned}
&P_{S0\to S0} = P_0 + P_1 \\
&P_{Si\to Si-1} = P_0 && \text{for } i = 1,2,\ldots,Q \\
&P_{Si\to Sj} = P_{j-i+1} && \text{for } i = 0,1,\ldots,Q-1,\ j = i,\ldots,Q \\
&P_{Si\to SQ} = \sum_{k=Q-i+1}^{\min(N,L)} P_k && \text{for } i = 0,1,\ldots,Q \\
&P_{Si\to Sj} = 0 && \text{otherwise}
\end{aligned}
\qquad \text{Eq. C.4}
$$

Assuming statistical equilibrium, where the probability of increasing state is equal to the probability of decreasing state at any state transition, leads to the Markov chain balance equations for figure C.2 shown in equation C.5.

$$P_{S1} \cdot P_0 = P_{S0} \cdot \sum_{k=2}^{\min(N,L)} P_k = P_{S0} \cdot (1 - P_0 - P_1)$$

$$\therefore \quad P_{S1} = \frac{1 - P_0 - P_1}{P_0} \cdot P_{S0}$$

$$P_{S2} \cdot P_0 = P_{S1} \cdot \sum_{k=2}^{\min(N,L)} P_k + P_{S0} \cdot \sum_{k=3}^{\min(N,L)} P_k$$

$$= P_{S1} \cdot (1 - P_0 - P_1) + P_{S0} \cdot (1 - P_0 - P_1 - P_2)$$

$$\therefore \quad P_{S2} = \frac{1 - P_1}{P_0} \cdot P_{S1} - \frac{P_2}{P_0} \cdot P_{S0}$$

$$\vdots$$

$$P_{SQ} \cdot P_0 = P_{SQ-1} \cdot \sum_{k=2}^{\min(N,L)} P_k + \cdots + P_{S0} \cdot \sum_{k=Q+1}^{\min(N,L)} P_k$$

$$\therefore \quad P_{SQ} = \frac{1 - P_1}{P_0} \cdot P_{SQ-1} - \sum_{k=2}^{Q} \frac{P_k}{P_0} \cdot P_{SQ-k} \qquad \text{Eq. C.5}$$

$$1 = \sum_{k=0}^{Q} P_{Sk} \quad \therefore \quad P_{S0} = \frac{1}{1 + \sum_{k=1}^{Q} \frac{P_{Sk}}{P_{S0}}}$$

The probability of a packet arriving at an output being lost due to queue overflow is given by the probability of no packet exiting the queue divided by the probability of one or more packets arriving at the output queue. The only way a packet will not exit the queue is if the queue is empty and no packets arrive. The probability of a packet arriving at the output queue is the probability of one or more packets arriving destined for the output scaled by the probability of the packets colliding and being dropped before they reach the output queue. The packet loss probability because of queue overflow is shown in equation C.6.

$$P_{\textit{arriving packet makes it through}} = 1 - P_{S0} \cdot P_0$$

$$P_{\textit{packet arrives}} = \frac{N \cdot \rho}{M} \cdot (1 - P_{\textit{collide}}) \qquad \text{Eq. C.6}$$

$$PLP_{\textit{queue overflow}} = 1 - \frac{1 - P_{S0} \cdot P_0}{\frac{N \cdot \rho}{M} \cdot (1 - P_{\textit{collide}})}$$

The total packet loss probability is the sum of the packet loss probability due to packet collisions and queue overflow as given in equation C.7.

$$PLP = \left(\frac{\sum_{i=L+1}^{N} (i-L) \cdot P_i}{\frac{N \cdot \rho}{M}} \right) + \left(1 - \frac{1 - P_{S0} \cdot P_0}{\frac{N \cdot \rho}{M} \cdot \left(1 - \frac{\sum_{i=L+1}^{N} (i-L) \cdot P_i}{\frac{N \cdot \rho}{M}} \right)} \right)$$

$$PLP = \left(\frac{\sum_{i=L+1}^{N} (i-L) \cdot P_i}{\frac{N \cdot \rho}{M}} \right) + \left(1 - \frac{1 - P_{S0} \cdot P_0}{\frac{N \cdot \rho}{M} \cdot \left(\frac{N \cdot \rho}{M} - \sum_{i=L+1}^{N} (i-L) \cdot P_i \right)} \right) \qquad \text{Eq. C.7}$$

Appendix D - Amplifier Noise Level Calculations

To calculate the noise performance of an electrical amplifier, the noise of the system used for the measurement must be determined. Simply attaching a matched load to the input of the test equipment used to measure the device under test allows this noise to be determined [HP75]. The HP 7000A spectrum analyzer requires a 50 Ω matched load. The theoretical noise generated by this load is given by equation D.1; any noise measured in excess of this must be added by the spectrum analyzer itself.

$$N_{i\,theory} = k \cdot T \cdot B$$
$$= 4.004 \cdot 10^{-21} \quad W/Hz$$
$$= -173.98 \quad dBm/Hz \qquad \text{Eq. D.1}$$
$$where: \quad k = 1.3807 \cdot 10^{-23} \quad J/K$$
$$T = 290 \quad K$$
$$B = 1 \quad Hz$$

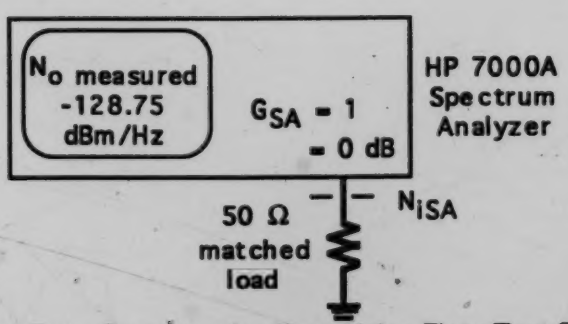


Figure D.1 - Spectrum Analyzer Noise Floor Test Setup

The actual noise measured on the spectrum analyzer at 1 GHz was -128.75 dBm/Hz using the test setup shown in figure D.1. Since the gain of the spectrum analyzer is 1 (0 dB), this noise level referred to the input of the spectrum analyzer (N_{iSA}) is also -128.75 dBm/Hz. The noise figure of the spectrum analyzer is calculated as in equation D.2.

$$NF_{7000A\,SA} = N_{i\,measured} - G_{7000A\,SA} - N_{i\,theory} \qquad \text{Eq. D.2}$$
$$= 45.225 \quad dB$$

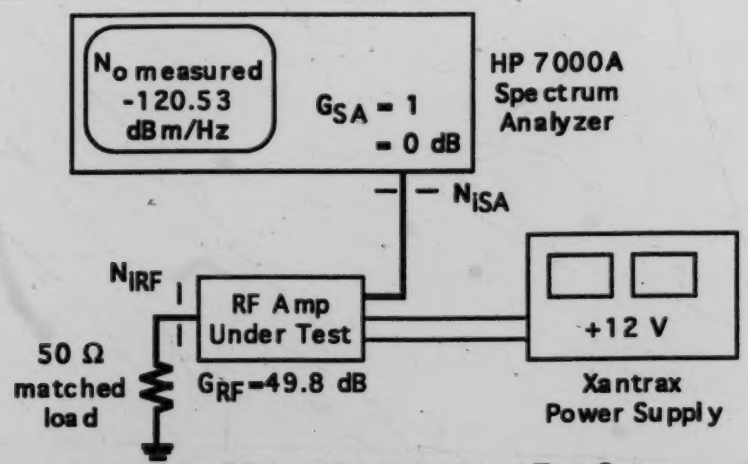


Figure D.2 - RF Amplifier Noise Level Test Setup

Inserting an RF amplifier as the device under test will increase the noise of the total system, with the noise figure of the RF amplifier alone being calculated by removing the spectrum analyzer excess noise. With the RF amplifier, the noise level was measured as -120.53 dBm/Hz (N_{iSA}) at 1 GHz, approximately 8 dBm/Hz worse than the spectrum analyzer itself. This test setup is shown in figure D.2. Referring this measured noise to the input of the RF amplifier, N_{iRF}, (with a gain of 49.8 dB at 1 GHz) results in a measured noise level referred to the input of the RF amplifier of -170.33 dBm/Hz. Using equation D.2 (with the gain set to 49.8 dB), this results in a noise figure for the total system under test (the RF amplifier and the spectrum analyzer) of 3.645 dB. This is a much smaller noise figure than for the spectrum analyzer alone because of the high gain of the RF amplifier.

To calculate the noise figure of the RF amplifier alone, the excess noise of the spectrum analyzer must be referred to the input of the RF amplifier and then subtracted from the excess noise of the total system as in equation D.3. The gain of the RF amplifier at 1 GHz was measured as 49.8 dB.

$$NF_{RF\,Amp} = 10 \cdot \log\left(10^{\frac{NF_{total}}{10}} - \frac{10^{\frac{NF_{7000A\,SA}}{10}} - 1}{10^{\frac{G_{RF\,amp}}{10}}} \right) \qquad \text{Eq. D.3}$$

$$= 2.936 \quad dB$$

The noise figure of the power amplifier cannot be calculated in exactly the same manner because the power amplifier has such a low gain that the excess noise from the spectrum analyzer swamps the noise added by the power amplifier and does not allow for an accurate measurement. To work around this problem, an RF amplifier was used as a pre-amplifier to the HP 7000A spectrum analyzer and then the calculations proceed in a very similar manner as above. First, the noise of the spectrum analyzer and RF amplifier together was measured as -119.48 dBm/Hz (N_{iSA}) at 1 GHz with an RF amplifier gain of 50.4 dB at 1 GHz using the test setup shown in figure D.3. Referring this measured noise to the input of the RF amplifier (N_{iRF}) using equation D.4 results in a measured noise figure of the spectrum analyzer and RF amplifier of 4.095 dB.

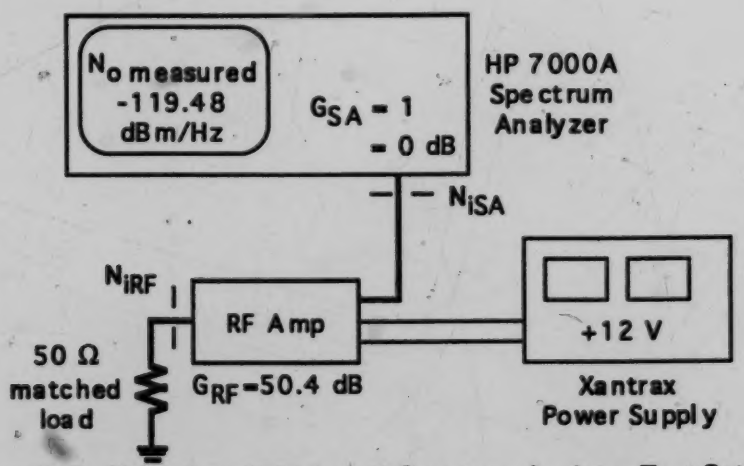


Figure D.3 - Noise Level RF Amp & Spectrum Analyzer Test Setup

$$NF_{SA\,\&\,RF\,amp} = N_{i\,measured} - G_{RF\,amp} - N_{i\,theory} \qquad \text{Eq. D.4}$$

$$= 4.095 \quad dB$$

Inserting the power amplifier as the first device, still using a 50 Ω matched load on the input as shown in figure D.4, and measuring the noise level allows the total system noise figure to be calculated using equation D.4 with the gain including the power amplifier gain as well. The power amplifier had a measured gain of 6.34 dB at 1 GHz, with the measured noise level of -110.02 dBm/Hz (N_{iSA}). This results in a total noise figure of 7.215 dB including the power amplifier, RF amplifier, and the spectrum analyzer. Using equation D.3, the noise figure of the power amplifier alone can be calculated as 6.904 dB.

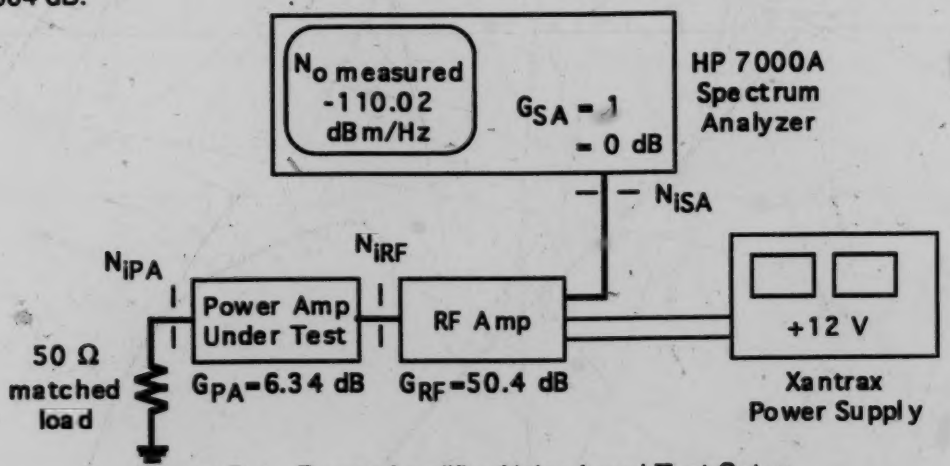


Figure D.4 - Power Amplifier Noise Level Test Setup